UNIVERSITÉ DU LUXEMBOURG

PhD-FSTM-2026-037

Faculty of Science, Technology and Medicine

DISSERTATION

Defence held on **20 March 2026** in Luxembourg
to obtain the degree of

DOCTEUR DE L'UNIVERSITÉ DU LUXEMBOURG EN INFORMATIQUE

by

Ali TOURANI
Born on 30 January 1991 in Iran

Thesis title:

Enhancing Robots' Situational Awareness using Imperceptible Artificial Landmarks

Dissertation Defence Committee:

Prof. Holger VOOS (Supervisor)
Full Professor in Engineering Science, SnT, University of Luxembourg, Luxembourg

Prof. Jan LAGERWALL (Chair)
Full professor in Experimental Polymer Physics, Department of Physics & Materials Science, University of Luxembourg, Luxembourg

Prof. Rafael MUNOZ-SALINAS
Full Professor in Computer Science, Department of Computing and Numerical Analysis, University of Cordoba, Spain

Dr. Martin R. OSWALD
Assistant Professor, Universiteit van Amsterdam, the Netherlands

Dr. Jose Luis SANCHEZ-LOPEZ
Research Scientist, SnT, University of Luxembourg, Luxembourg

Affidavit / Statement of originality

I, **Ali TOURANI**, declare that this thesis:

- is the result of my own work. Any contribution from any other party, and any use of generative artificial intelligence technologies, have been duly cited and acknowledged;
- is not substantially the same as any other that I have submitted, and;
- is not being concurrently submitted for a degree, diploma, or other qualification at the University of Luxembourg or any other University or similar institution except as specified in the text.

With my approval, I furthermore confirm the following:

- I have adhered to the rules set out in the University of Luxembourg's Code of Conduct and the Doctoral Education Agreement (DEA), in particular with regard to Research Integrity.
- I have documented all methods, data, and processes truthfully and fully.
- I have mentioned all the significant contributors to the work.
- I am aware that the work may be screened electronically for originality.

I acknowledge that if any issues are raised regarding good research practices in the review of the thesis, the examination may be postponed pending the outcome of any investigation into such issues. If a degree was conferred, any such subsequently discovered issues may result in the cancellation of the degree.

Approved on 2026-02-13, Luxembourg

Abstract

Fiducial markers have long served as highly distinguishable *geometric* and *visual anchors* for establishing correspondences between 3D world points and their 2D image projections. Considering their unique features, they are widely used across augmented and mixed reality, computer vision, and robotics. In robotics specifically, fiducial markers play a central role in visual sensor **calibration**, sensor **synchronization**, **Human-Robot Interaction**, and **Visual SLAM (VSLAM)**. As robotic systems grow more capable and operate in increasingly diverse environments, these information-rich artificial landmarks remain a practical and reliable means of improving robustness, interpretability, and overall system performance. However, despite these advantages, a fundamental challenge emerges when envisioning environments populated with numerous fiducial markers. If every surface were required to host visually prominent markers, the environment would rapidly become cluttered, intrusive, and aesthetically undesirable for human occupants. This trade-off between the **visibility** required for robot-friendly perception and the **unobtrusiveness** expected in human-centric spaces forms the central motivation of this thesis.

Accordingly, this thesis investigates the development of **iMarkers**, unobtrusive, non-distracting, and ideally invisible fiducial markers that remain reliably detectable by robots. The central idea behind iMarkers is to replace traditional printed pigments with microscopic Cholesteric Spherical Reflector (CSR) shells, enabling information to be embedded directly onto surfaces without altering their human-facing appearance. The primary goal of the thesis is to design, employ, and evaluate these imperceptible markers and to determine whether they can support enhanced **robotic situational awareness** by providing semantic and geometric cues without compromising the aesthetics of real-world environments. To address this, the author has appended **five contributed papers** that span fabrication considerations, detector sensor designs, the corresponding implemented algorithms, and their potential contributions to robots' **situational awareness**.

As a concrete robotics case study, the thesis first investigates how **classic fiducial markers** can be elevated to support higher-level semantic reasoning. By placing printed markers on indoor environments' walls, a **marker-aware Visual SLAM** system is developed that converts simple geometric cues into meaningful spatial entities, such as walls, corridors, and rooms. This demonstrates that even lightweight, non-learning-based signals can enrich a robot's understanding of its environment. Building on this, the thesis introduces **hierarchical scene graphs** that organize keyframes, map points, markers, and structural components into a multi-layer representation. This hierarchy enables the generation of human-interpretable **digital twins**, bridging the gap between raw sensor data and structured spatial knowledge. The thesis then presents the first **robotics-oriented integration of iMarkers**, showing that these imperceptible fiducial markers can be detected, decoded, and semantically assigned within a VSLAM pipeline. Despite being available only in small quantities during fabrication, the experiments show that iMarkers offer the same semantic utility as classical markers, validating their use for unobtrusive mapping

and augmented indoor environments. Finally, all earlier insights converge into the thesis's main contribution: **vS-Graphs**, a robust, scalable, and marker-optional RGB-D VSLAM framework. vS-Graphs tightly couples visual perception, geometric SLAM, and semantic scene understanding. It introduces parallel threads for detecting *building components* (walls and ground surfaces) and inferring **structural elements** (rooms and floors), and integrates them into a unified, optimizable 3D scene graph. This representation provides robots with rich, consistent, and actionable situational awareness—whether or not artificial landmarks are present—paving the way toward more transparent, adaptive, and human-aligned environment modeling.

Overall, this thesis shows that robots can attain robust **situational awareness** by integrating *geometric reasoning*, *semantic understanding*, and *artificial cues* such as iMarkers. The proposed methods contribute to more adaptive, explainable, and human-aligned representations of the environment, ultimately improving environment modeling and digital twin generation in robotics.

Acknowledgements

This thesis would not have been possible without the support, guidance, and encouragement of many people, including my supervisory team, colleagues, friends, and family. First and foremost, I would like to express my deepest gratitude to **Prof. Holger Voos**, my supervisor, for his continuous support, availability, and guidance throughout this journey. There is no doubt that his commitment and constructive feedback were fundamental to the development and completion of this thesis. I am also sincerely grateful to **Dr. Jose Luis Sanchez-Lopez**, whose insights and expertise in the domain of robotics situational awareness have significantly shaped the direction of my research. He was always open to discussing how to create meaningful impact in the robotics community and generously supported me throughout every publication. Additionally, I am truly fortunate to have worked with **Prof. Jan PF Lagerwall**, whose interdisciplinary perspective and vision for environments enriched with invisible augmented information significantly shaped my thinking throughout this journey. I would also like to thank **Prof. Rafael Muñoz Salinas** for his insightful help, support, and suggestions throughout this journey

At the collaboration level, I am deeply grateful to **Dr. Deniz Işınsu Avşar**, with whom I had the privilege of shaping an interdisciplinary Ph.D. pathway across two distinct research domains. Bridging two different backgrounds and mindsets is challenging, but her support and friendship made the journey both enjoyable and rewarding. I truly appreciate her dedication to the iMarker fabrication efforts and her commitment to our joint work within the **TRANSCEND** project. I am also thankful to the **Institute for Advanced Studies (IAS)** and, in particular, to **Prof. Claus Vögele** and **Dr. Sylvie Fromentin** for their continuous efforts to foster interdisciplinary research through workshops, meetings, and social activities. Their continuous financial and organizational support provided an environment where collaboration felt both pleasant and manageable.

Special thanks go to all my colleagues in the **Automation and Robotics Group (ARG)**, especially my friends and teammates in the situational awareness team, with whom I shared an unforgettable four-year journey. Although the team evolved over time, with members leaving and new ones joining, it was always a pleasure to work together and build something meaningful. When I first joined ARG in April 2022, the core foundations of **S-Graphs** were beginning to take shape, and I became responsible for its vision-based component and version. Looking back, it is inspiring and satisfying to see how much the core concept has grown, broadened, and matured through everyone's contributions, where now it covers a wide range of versions (multi-robot, reasoning-based, learning-based, dynamic, informed, mixed-reality-based, and more to come). I want to extend my gratitude to all the master's and bachelor's students I had the pleasure of advising and mentoring throughout these years, as well as to everyone at *AeroLab*. Their energy and dedication enriched my daily work and made my research experience even more rewarding.

Finally, I want to express my heartfelt gratitude to my **family** and **friends** for their continuous support, encouragement, patience, and understanding throughout this journey.

Pursuing a Ph.D. and starting a new chapter in life is highly challenging, and having a circle filled with love, kindness, support, and joy made all the difference. My family has always been my source of strength, offering emotional support through every high and low. They have been my role models, and I hope I have been able to return at least a fraction of the happiness they have given me. I also feel incredibly fortunate to have met my fiancée during this period. Her support, love, care, and presence, along with all the journeys we discovered together, have greatly enriched my life and brought fulfillment to these challenging years. Additionally, I was lucky to be surrounded by many close friends whom I met at SnT and who have since become like brothers and sisters to me. Without the support of all these wonderful people, completing this thesis would not have been possible. I wish to conclude this acknowledgments section with my favorite Ghazal by *Hafez Shirazi*, the renowned Persian lyric poet, which deeply reflects my life aspirations:

ده‌روزه مهر گردون افسانه است و افسون
نیکی به جای یاران فرصت شمار یارا

The world lasts as a ten-day story, a fable, a fleeting spell;
Then, friend, count each moment as a chance to show kindness to real companions.

ای صاحبِ کرامت شُکرانهٔ سلامت
روزی تفقُّدی کن درویشِ بی‌نوا را

O noble one, as a token of gratitude for your well-being,
Once in a while, cast a glance at the destitute people.

آسایشِ دو گیتی تفسیر این دو حرف است
با دوستان مروّت با دشمنان مدارا

The comfort of the world is found in interpreting this simple counsel:
With friends, magnanimity; with foes, patience and forbearance.

Contents

I Comprehensive Summary **1**

1 Introduction **2**
1.1 Preface . . . 2
1.2 Background Information . . . 3
1.2.1 Light and Optics Concepts . . . 3
1.2.2 Liquid Crystals . . . 5
1.2.3 Fiducial Markers . . . 7
1.2.4 Fiducial Markers in Robotics . . . 8
1.3 Literature Review . . . 11
1.3.1 Classic Fiducial Markers . . . 11
1.3.2 Blended and Unobtrusive Fiducial Markers . . . 15
1.3.3 Visual SLAM Systems . . . 16
1.3.4 Marker-based Visual SLAM . . . 19
1.3.5 High-Level Scene Representation . . . 21
1.4 Research Motivation & Objectives . . . 22
1.4.1 Research Motivation . . . 22
1.4.2 Research Objectives . . . 24
1.5 Significance of the Study . . . 24
1.6 Thesis Structure . . . 25

2 Contributions **27**
2.1 Detection and Integration of iMarkers (RQ1) . . . 27
2.1.1 Overview . . . 27
2.1.2 Contributions . . . 28
2.1.3 Summary of Results . . . 31
2.2 State-of-the-art Visual SLAM Approaches (RQ2) . . . 32
2.2.1 Overview . . . 32
2.2.2 Contributions . . . 32
2.2.3 Summary of Results . . . 33
2.3 Marker-Enhanced Perception (RQ3) . . . 35
2.3.1 Overview . . . 35
2.3.2 Contributions . . . 35
2.3.3 Summary of Results . . . 37
2.4 iMarker-integrated VSLAM (RQ4) . . . 38
2.4.1 Overview . . . 38
2.4.2 Contributions . . . 38
2.4.3 Summary of Results . . . 40

2.5 Minimal-Landmark Situational Awareness (RQ5) . . . 41
2.5.1 Overview . . . 41
2.5.2 Contributions . . . 41
2.5.3 Summary of Results . . . 43

3 Conclusions **45**
3.1 Limitations and Open Challenges . . . 45
3.1.1 iMarker Integration into Robotics . . . 45
3.1.2 Situational Awareness Framework . . . 47
3.2 Future Research Directions . . . 47
3.2.1 Smartphone-Compatible iMarker Detection . . . 47
3.2.2 Supporting Additional Semantic Entities . . . 48
3.2.3 vS-Graphs Scaling . . . 48
3.2.4 Reasoning-based Structural Element Recognition . . . 49
3.2.5 Semantic Loop Closure Detection in vS-Graphs . . . 49
3.2.6 Incorporating Architectural Priors in vS-Graphs . . . 49
3.2.7 Exploring Vision-Language Models . . . 49
3.3 Closing Insights . . . 50

II Appended Articles **51**

4 iMarker Detection **52**
4.1 Abstract . . . 53
4.2 Introduction . . . 53
4.3 Literature Review . . . 54
4.3.1 Ordinary Fiducial Markers . . . 54
4.3.2 Blended and Unobtrusive Fiducial Markers . . . 55
4.4 iMarkers: Concept and Comparison . . . 56
4.5 Sensor Design and Detection Strategies . . . 57
4.5.1 Dual-vision Solution . . . 57
4.5.2 Dynamic Single-vision Setup . . . 59
4.5.3 Static Single-vision Setup . . . 60
4.6 Benchmarking and Evaluation . . . 62
4.6.1 Operational Implementation . . . 62
4.6.2 Qualitative Evaluation . . . 63
4.6.3 Detection Range and Pose Accuracy Evaluation . . . 64
4.6.4 Operation in Degraded Visual Conditions . . . 67
4.6.5 Recognition Speed . . . 69
4.7 Discussions . . . 70
4.7.1 Comparative Insights . . . 70
4.7.2 Potential Applications . . . 72
4.7.3 Limitations and Fabrication Considerations . . . 72
4.8 Conclusions . . . 74

5 Visual SLAM Survey **75**
5.1 Abstract . . . 76
5.2 Introduction . . . 76
5.3 Related Surveys . . . 78

5.4 Evolution of Visual SLAM Algorithms . . . 79
5.5 VSLAM Setup . . . 81
5.5.1 Sensors and Data Acquisition . . . 81
5.5.2 Visual Features . . . 81
5.5.3 Target Environments . . . 82
5.5.4 System Evaluation . . . 82
5.5.5 Semantic Level . . . 83
5.6 VSLAM State-of-the-Arts . . . 85
5.6.1 Objective I: Multi-Sensor Processing . . . 85
5.6.2 Objective II: Pose Estimation . . . 87
5.6.3 Objective III: Real-World Viability . . . 89
5.6.4 Objective IV: Resource Constraint . . . 92
5.6.5 Objective V: Versatility . . . 93
5.6.6 Objective VI: Visual Odometry . . . 94
5.7 Current Trends . . . 97
5.7.1 Statistics . . . 97
5.7.2 Analyzing the Current Trends . . . 99
5.8 Conclusions . . . 100

6 Marker-Based Visual SLAM **102**
6.1 Abstract . . . 103
6.2 Introduction . . . 103
6.3 Related Works . . . 105
6.4 Proposed Method . . . 106
6.4.1 Overview . . . 107
6.4.2 Semantic Entities . . . 107
6.4.3 Final Graph . . . 109
6.5 Evaluation . . . 109
6.5.1 Evaluation Setup . . . 109
6.5.2 Experimental Results . . . 110
6.6 Conclusions . . . 112

7 Marker-Driven Semantics **114**
7.1 Abstract . . . 115
7.2 Introduction . . . 115
7.3 Related Works . . . 117
7.3.1 SLAM and 3D Scene Graphs . . . 117
7.3.2 Fiducial Markers and Marker-Based SLAM . . . 118
7.4 Proposed Method . . . 120
7.4.1 Fundamentals . . . 122
7.4.2 Structural-Level and Higher-Level Semantic Entities . . . 122
7.4.3 The Final S-Graph . . . 125
7.4.4 Inclusion of iMarkers . . . 125
7.5 Evaluation . . . 126
7.5.1 Evaluation Setup . . . 126
7.5.2 Experimental Results . . . 127
7.6 Discussions . . . 129
7.7 Conclusions . . . 130

8 Visual S-Graphs (vS-Graphs) 132
8.1 Abstract . . . 133
8.2 Introduction . . . 133
8.3 Related Works . . . 135
8.4 Proposed Method . . . 136
8.4.1 System Overview . . . 136
8.4.2 Building Component Recognition . . . 137
8.4.3 Structural Element Recognition . . . 138
8.4.4 Scene Graph Structure . . . 140
8.5 Experimental Results . . . 140
8.5.1 Methodology . . . 140
8.5.2 Evaluations and Discussions . . . 141
8.6 Conclusions . . . 146
8.7 Appendices . . . 146

III Appendices 150

9 Acronyms 151

10 Author's Research Contributions 154

11 Open-Source Software Contributions 157

List of Figures

1.1 Visible light spectrum . . . 3
1.2 Polarization variants and impacts . . . 4
1.3 Beamsplitter functionality . . . 5
1.4 The impact of viewing angles and polarization on CSRs . . . 7
1.5 Fiducial Marker recognition steps . . . 8
1.6 Sample fiducial marker applications in robotics . . . 9
1.7 The impact of fiducial markers in VSLAM . . . 10
1.8 Classic fiducial marker libraries . . . 14
1.9 Blended fiducial markers . . . 17
1.10 ORB-SLAM variants architecture . . . 18
1.11 Marker-based Visual SLAM frameworks . . . 20
1.12 3D Scene Graph Representation Methods . . . 21
1.13 An environment augmented with many markers . . . 23
1.14 Thesis structure . . . 25

2.1 Fabricating iMarkers with CSRs (Paper-1) . . . 28
2.2 ArUco marker dictionary variants . . . 29
2.3 Variant iMarker detection sensors (Paper-1) . . . 30
2.4 Maximum detection distances for printed and iMarkers (Paper-1) . . . 31
2.5 The trend of VSLAM methodologies (Paper-2) . . . 34
2.6 Semantic UcoSLAM pipeline (Paper-3) . . . 36
2.7 Semantic UcoSLAM map quality (Paper-3) . . . 37
2.8 Semantic ORB-SLAM 3.0 (Paper-4) . . . 39
2.9 Semantic ORB-SLAM 3.0 evaluation (Paper-4) . . . 40
2.10 An environment map generated by vS-Graphs (Paper-5) . . . 42
2.11 vS-Graphs qualitative results (Paper-5) . . . 43

4.1 Augmenting environments with iMarkers . . . 54
4.2 Fabricating iMarkers with CSRs . . . 57
4.3 Versatility in iMarker production . . . 58
4.4 Dual-vision setups for iMarker detection . . . 59
4.5 Single-vision setups for iMarker detection . . . 61
4.6 Variant iMarker detection sensors . . . 64
4.7 iMarker variants detected using different sensor setups . . . 65
4.8 Maximum detection distances for printed and iMarkers . . . 67
4.9 Relative performance scores of iMarker variants . . . 71
4.10 Real-world robotics applications enabled by iMarkers. . . . 73

5.1 The flowchart of a standard visual SLAM approach . . . 78
5.2 Milestones of impactful VSLAM approaches . . . 80

5.3 Popular VSLAM datasets . . . 84
5.4 Current trends of VSLAM methods . . . 98

6.1 A scene graph generated by Semantic UcoSLAM . . . 104
6.2 Semantic UcoSLAM pipeline . . . 106
6.3 Semantic UcoSLAM scene graph structure . . . 110
6.4 In-house collected dataset with ArUco markers . . . 111
6.5 Semantic UcoSLAM maps quality against SOTA . . . 113
6.6 ATE of Semantic UcoSLAM compared to the baseline . . . 113

7.1 Overall idea of marker-based Semantic ORB-SLAM 3.0 . . . 117
7.2 A transparent iMarker on a door frame . . . 119
7.3 The pipeline of the proposed Semantic ORB-SLAM 3.0 . . . 121
7.4 Scene graph structure of the proposed framework . . . 125
7.5 In-house dataset for evaluating the proposed method . . . 127
7.6 ATE evaluation of the proposed Semantic ORB-SLAM 3.0 . . . 128

8.1 An environment map generated by vS-Graphs . . . 134
8.2 vS-Graphs architecture . . . 136
8.3 vS-Graphs scene graph structure . . . 141
8.4 AutoSense device for in-house data collection . . . 142
8.5 Mapping performance of vS-Graphs and ORB-SLAM 3.0 . . . 144
8.6 Qualitative results of scene graph generation . . . 145
8.7 Timeline of thread execution within vS-Graphs . . . 146
8.8 Infusing fiducial marker recognition in vS-Graphs . . . 147
8.9 The role of fiducial markers within vS-Graphs . . . 148
8.10 Qualitative results of scene reconstruction using vS-Graphs . . . 149

List of Tables

1.1 Beamsplitter variants based on construction 6

2.1 iMarkers detection and recognition performance (Paper-1) 32

4.1 Unobtrusive fiducial markers 56
4.2 Mean pose estimation errors of iMarkers 66
4.3 Detection of printed markers and iMarkers under low illumination 68
4.4 iMarkers detection and recognition performance 69

5.1 Commonly used VSLAM datasets 85

6.1 Collected indoor dataset's characteristics 111
6.2 RMSE & STD of Semantic UcoSLAM and baseline 112

7.1 Collected dataset characteristics 126
7.2 Evaluation results using RMSE in meters 129

8.1 Characteristics of the *AutoSense* in-house dataset 141
8.2 Performance of vS-Graphs against state-of-the-art frameworks 143
8.3 Scene understanding performance of vS-Graphs 144

Part I

Comprehensive Summary

1

Introduction

1.1 Preface

This thesis is part of an interdisciplinary project titled **TRansforming Autonomous Navigation, Swarm robotics and Construction by ENcoding Data into surfaces** (TRANSCEND), funded by the Institute for Advanced Studies (IAS) of the University of Luxembourg. The project was conducted during my Ph.D. studies in Computer Science, in close collaboration with a fellow Ph.D. researcher in the Department of Physics and Materials Science, and under the joint supervision of two faculty advisors. The overarching goal of TRANSCEND was to establish a state-of-the-art framework for designing and deploying a new generation of fiducial markers, referred to as *iMarkers*, and to demonstrate their potential application in the robotics domain. The project introduces a novel paradigm in robotic perception: augmenting environments with **low-visibility, information-rich artificial landmarks** that enhance a robot's situational awareness without introducing significant visual pollution. This thesis examines the feasibility and impact of this concept from the robotics perspective, providing a practical use case that evaluates performance improvements in real-world scenarios. As the project spans multiple research domains, including robotics, optics, material science, and Human–Robot Interaction (HRI), this chapter provides the necessary background and conceptual foundations that will be referenced in the subsequent chapters of the thesis.

Scope Clarification

Given the interdisciplinary nature of this project, a concise overview of the employed optical components and material science concepts is necessary. However, the thesis does not aim to explore domain-specific theories in depth; instead, it focuses only on the concepts required to support the proposed methodologies and experiments.

1.2 Background Information

1.2.1 Light and Optics Concepts

As shown in Figure 1.1, visible light spans wavelengths of approximately $380-750$ nm, positioned between **Ultraviolet (UV)** radiation at shorter wavelengths and **Infrared (IR)** radiation at longer wavelengths [1]. Although other forms of electromagnetic radiation, such as gamma rays, X-rays, and radio waves, can also be considered types of light, the term *light* in this thesis refers exclusively to **visible light**. Accordingly, the study of light, known as *optics*, is a fundamental discipline within modern physics that investigates the behavior of light, its interactions with materials, and the mechanisms used for its detection [2].

When discussing electromagnetic waves, the concept of *polarization* arises in describing how the wave behaves as a two-dimensional transverse oscillation. Light consists of synchronized oscillating *electric* and *magnetic* fields, which are always perpendicular to each other and to the direction of propagation [3]. Thus, polarization refers explicitly to the orientation of the *electric field* component, indicating the direction in which this field vector oscillates. A typical example of polarization in everyday applications is found in 3D cinema glasses (also known as 3D goggles), which use circular polarization. These goggles are designed so that the right lens transmits only right-handed circularly polarized light, while the left lens transmits only left-handed circularly polarized light. As a result, each eye receives a slightly different image, allowing the viewer to perceive depth and experience a 3D visual effect.

There are three primary forms of polarization in optics [3], as illustrated in Figure 1.2a, including:

- **Linear polarization**: The electric field oscillates in a single, fixed direction within the plane perpendicular to the direction of propagation.
- **Elliptical polarization**: The tip of the electric field vector traces an ellipse over time as the wave propagates, mainly when the horizontal and vertical components of the electric field have different amplitudes. Circular polarization is a special case in which the electric field components have equal amplitude and a 90° phase difference, causing the electric field vector to rotate uniformly.
- **Unpolarized light**: The electric field oscillates randomly in all directions perpendicular to the direction of propagation, with no fixed polarization state over time.

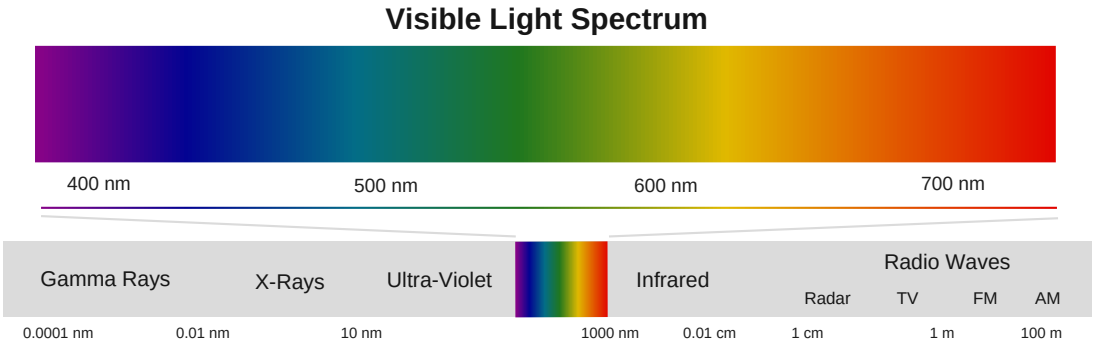
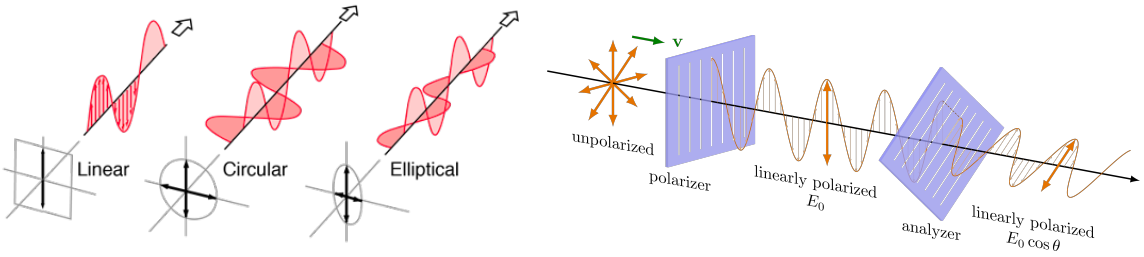


Figure 1.1: Relative position of the visible light spectrum within the full range of electromagnetic radiation.



(a) Polarization variants (b) Polarization impact

Figure 1.2: Different types of light polarization and their effects: **(a)** schematic representations of linear, elliptical, and circular polarization[1]; **(b)** illustration of how polarization influences light behavior and interaction with materials[2].

Moreover, light can exhibit **right-hand (clockwise) polarization**, where the electric field rotates in a right-hand sense relative to the direction of wave propagation, or **left-hand (counterclockwise) polarization**, where the rotation follows a left-hand sense.

Natural light exhibits random and time-varying polarization states in three-dimensional space and is therefore referred to as *unpolarized* or *randomly polarized* light. Hence, the electric field can oscillate horizontally, vertically, or in any direction between the two. By using optical filters known as **polarizers**, only the component of the light that oscillates in a specific direction is transmitted while all other elements are absorbed. In other words, a polarizer allows light to pass only along its defined polarization axis, as illustrated in Figure 1.2b. As shown, a *linear polarizer* can reduce glare from reflected light and convert unpolarized incoming light into linearly polarized light by restricting the electric field oscillation to a single plane. As a sample use case, linear polarizers are commonly used in digital imaging to reduce reflections and glare, particularly when capturing scenes with highly reflective surfaces or at bright illumination angles.

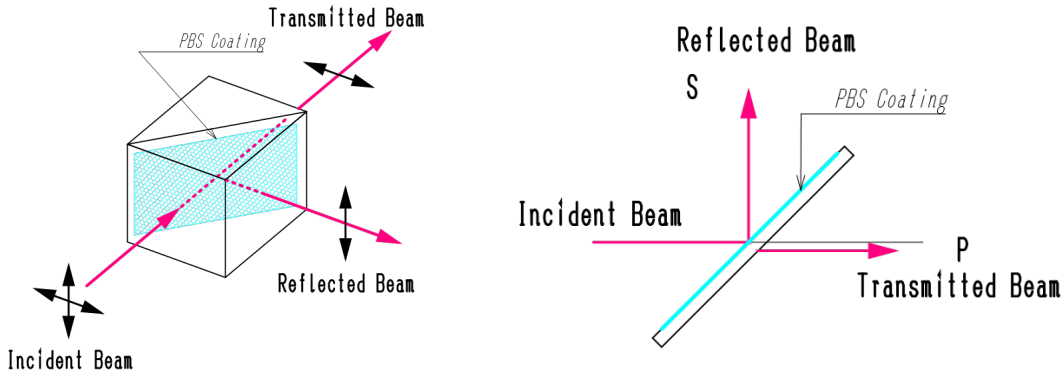
While **polarizers** control the orientation of light waves, another class of optical elements known as **beamsplitters** manipulates the distribution of light by dividing it into multiple paths. A beamsplitter can either *combine* two separate beams into one or *split* an incoming beam into two outputs by partially transmitting and partially reflecting the incident light [4]. The division can be defined by a *specified ratio*, which influences the resulting intensity and, depending on the design, may affect the wavelength or polarization state of the output beams. Figure 1.3 illustrates the splitting process in a cube beamsplitter.

Beamsplitters can be categorized in several ways, including by construction and polarization interaction. In this thesis, we primarily consider *standard beamsplitters*, which divide an incoming beam according to a fixed transmission–reflection ratio without altering or depending on the beam's polarization state. However, it is worth mentioning that other variants also exist, such as *polarizing* beamsplitters (which separate an unpolarized beam into two orthogonal linear components), *non-polarizing* beamsplitters (which split by intensity while preserving polarization), and *dichroic* [5]. In terms of construction, beamsplitters can be categorized into two primary types [6], which are summarized in Ta-

[1]Image taken from `http://hyperphysics.phy-astr.gsu.edu/hbase/phyopt/polclas.html`, accessed on 10 November 2025.

[2]Image taken from `https://tikz.net/optics_polarization/`, accessed on 10 November 2025.

[3]Image taken from `https://www.koshin-kogaku.com/application-notes/an-introduction-to-beam-splitter/`, accessed on 10 November 2025.



(a) 3D view (b) 2D view

Figure 1.3: The procedure of splitting the incident light into two beams using a beamsplitter[3].

ble 1.1. Accordingly, a cube beamsplitter is formed by cementing two right-angle prisms together, with the surface of one prism coated. In contrast, a plate beamsplitter consists of a thin, flat glass plate designed for a 45° Angle of Incidence (AOI), typically coated on the first surface with an anti-reflection coating applied to the second surface to minimize unwanted reflections[6].

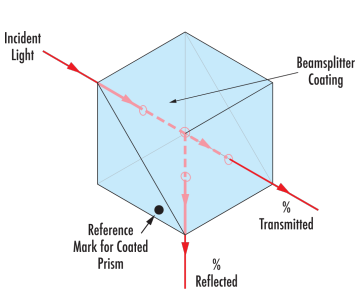
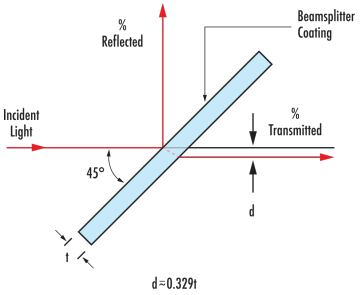
Apart from construction and polarization variants, several additional factors influence the selection of a beamsplitter. The **coating** determines its suitability across spectral ranges such as UV, visible, or IR, and helps minimize unwanted reflections and ghost images. The **Reflection-to-Transmission (R/T)** ratio, commonly 50:50, defines how the incident light is divided between reflected and transmitted paths. The physical **dimensions** vary depending on shape (*e.g.,* rectangular, square, or circular) and are typically specified in millimeters. Finally, the substrate **material** is chosen according to the intended application, as different materials offer distinct optical and mechanical properties.

1.2.2 Liquid Crystals

Liquid Crystals (LCs) are unique materials that exhibit characteristics of *liquids* and *solids*, enabling them to flow slowly like liquids while maintaining an ordered molecular structure typical of solids [7]. Due to their responsive and self-organizing nature, liquid crystals play a crucial role in various information technologies, most notably in the development of flat-panel displays [8]. A specific subclass, known as **Cholesteric Liquid Crystals (CLCs)**, exhibits selective reflection of light at particular wavelengths and polarizations. When formed into spherical shells, CLCs can selectively reflect light *omnidirectionally*, with tunable reflection across the UV to IR spectrum. Consequently, viewing a CLC sample through a circular polarizer of opposite handedness significantly suppresses the reflected light, demonstrating its polarization-dependent optical response.

Building upon the optical properties of CLCs, a specialized implementation known as **Cholesteric Spherical Reflectors (CSRs)** can achieve unique reflection characteristics. CSRs are distinguished by their narrow wavelength selectivity and circular polarization response [8]. They exhibit *omnidirectional retro-reflectivity*, meaning they can reflect incident light toward the source from any direction, even under complex illumination conditions [9]. It should be noted that CSRs are passive elements and do not emit light but rely on external illumination sources for visibility under different environmental conditions. Unlike conventional retro-reflective materials such as road signs or

Table 1.1: Comparison of beamsplitters based on their construction [6].

	Cube Beamsplitter	**Plate Beamsplitter**
Shape		
Construction	Two right angle prisms	A thin, flat glass plate
Lightweight?	✗	✓
Beam shift?	✗	✓
Ghost image?	✗	✓
Alignment	No need	Hard (*e.g.*, 45°)
Customization	Hard	Easy
Relative price	High	Low

high-visibility clothing, which reflect light broadly and without polarization specificity, CSRs can be engineered to reflect within defined wavelength bands and with a chosen handedness (left- or right-circular polarization) [10].

The fabrication of CSR shells is achieved by generating and flowing microscopic particles (typically $100-200\ \mu$m) from a controlled liquid crystal mixture. When **unpolarized** white light illuminates CSRs from different angles, the reflected light becomes **circularly polarized** and confined to a specific, narrow wavelength band [9]. This unique optical property enables the production of shells that reflect distinct colors across different regions of the spectrum, including UV, visible, and IR, depending on the design parameters of the liquid crystal structure [11]. Another remarkable feature of CSRs is their **polarization selectivity**, meaning that the reflected colors are circularly polarized and can appear either visible or invisible when viewed through right- or left-handed polarizers. As shown in Figure 1.4, polymerized CSRs observed under different polarization states exhibit distinct optical behaviors. Thus, particular **color spectra** are visible only under right-handed polarization, while others appear exclusively under left-handed polarization. Furthermore, in contrast with CLCs, they reflect visible light within a narrow band centered around a wavelength that does not vary with the **viewing angle** shift [12].

Given their unique optical properties, CSRs have a broad range of potential applications across domains such as robotics, security, and medicine. Their primary advantages include omnidirectional reflectivity, tunable and selective visibility, and the ability to produce highly distinctive patterns. For instance, in the security domain, they can enable authentication of physical or digital artifacts through token-based mechanisms alongside biometric methods [13]. Furthermore, CSRs can serve as advanced landmarks for robotic and Augmented Reality (AR) systems, encoding rich visual information that remains visible only to specific, verified sensors while invisible to others [8]. This particular application domain of CSRs shapes the primary focus of this thesis.

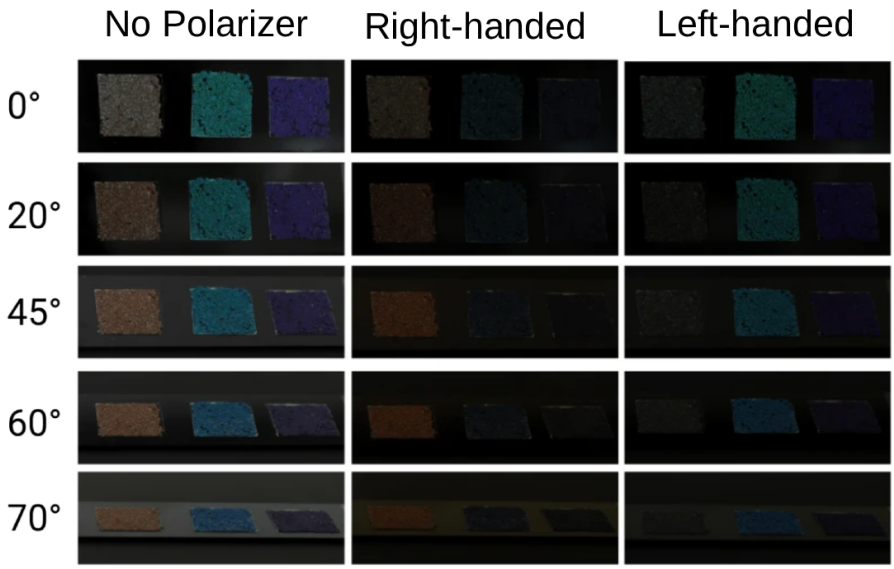


Figure 1.4: Polymerized CSR films under varying viewing angles and polarization states [12]. The reflected color does not shift with viewing angle, and distinct color spectra are observed for right-handed and left-handed polarizations.

1.2.3 Fiducial Markers

Fiducial markers (or simply markers) play a crucial role in various domains, including computer vision, AR, robotics, and Human–Robot Interaction (HRI), by serving as **artificial reference patterns** that enable reliable detection, localization, and pose estimation within a scene [14]. They provide well-defined *geometric* and *visual anchors* that help establish accurate correspondences between 3D world points and their two-dimensional image projections. Fiducial markers are designed with highly distinguishable *visual patterns* that are more robust and reliable than naturally occurring environmental features. This makes them particularly valuable for applications that require consistent feature matching, accurate pose estimation, and stable interaction with the physical world [15]. In challenging scenarios, such as textureless environments or uniformly colored surfaces like white walls, where conventional visual features are scarce, augmenting the scene with fiducial markers provides dependable landmarks for computer vision–based systems.

While QR codes [16] are primarily designed for data encoding and rapid scanning, they can also function as fiducial-like markers in computer vision by providing easily identifiable patterns for detection. Fiducial markers, however, extend this functionality by enabling accurate pose estimation [17]. Their generation and detection rely on specific computer vision libraries that utilize the camera's **intrinsic parameters** (*e.g.,* focal length and distortion coefficients) and pattern recognition procedures to accurately estimate the marker's position, orientation, and scale [18].

The general procedure for fiducial marker detection and pose estimation is depicted in Figure 1.5, where the marker detection library analyzes the camera-captured input frames. Each frame is first converted to grayscale and smoothed to reduce noise, then undergoes basic image processing operations, such as edge detection and contour extraction, to identify square shapes or predefined marker patterns. Once candidate regions are found, thresholding and binary segmentation are applied to isolate high-contrast areas that match the expected marker geometry. Subsequently, perspective correction is applied to rectify the candidate regions and account for camera viewpoint distortions. Each candidate

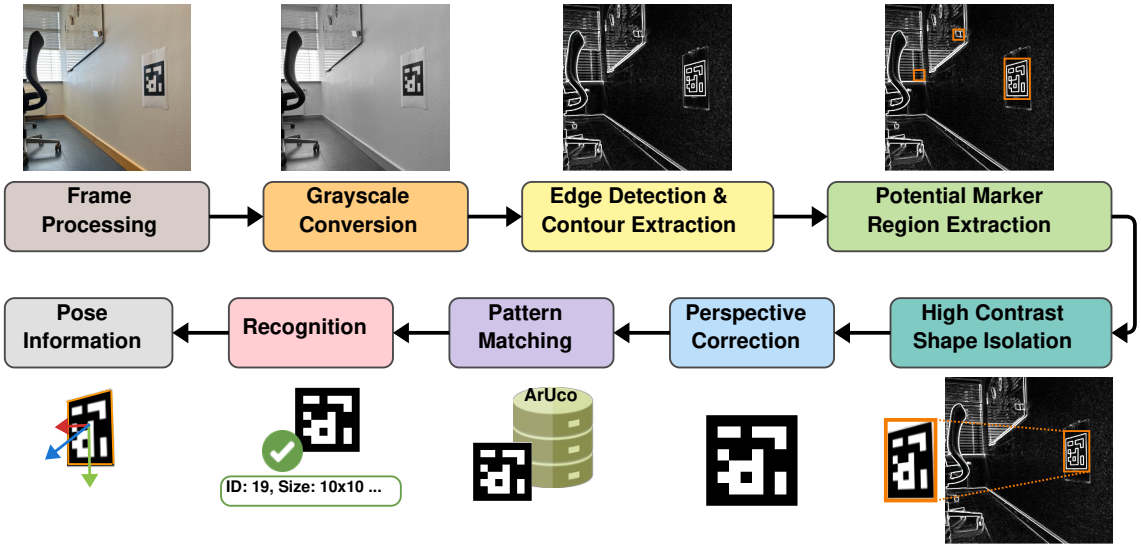


Figure 1.5: General detection and pose estimation pipeline for fiducial markers, illustrating primary steps from image acquisition and marker identification to 3D pose estimation.

region is then compared against a dictionary of marker patterns, allowing the library to decode the marker's identity (*i.e.,* marker_id) based on its internal binary pattern. Accordingly, only valid fiducial marker candidates will remain, and the rest are filtered out. The final stage, following successful detection and identification, is *pose estimation*, in which the library determines the marker's spatial position and orientation relative to the camera. This is achieved by solving a **Perspective-n-Point (PnP)** problem [19], which involves estimating the camera's **extrinsic parameters** (*i.e.,* rotation and translation) given the known 3D coordinates of the marker's corners and their corresponding 2D image projections. These parameters define the marker's 3D pose (position and orientation) relative to the camera's coordinate frame.

1.2.4 Fiducial Markers in Robotics

Beyond their widespread applications in AR and Mixed Reality (MR) systems, where they enable virtual content to be overlaid onto physical environments, fiducial markers are reliable tools across a broad range of robotic applications, which form the central focus of this thesis. As depicted in Figure 1.6, markers are embedded into numerous perception, control, and interaction pipelines of modern robotic systems. A fundamental role of fiducial markers lies in **visual sensor calibration**, where their stable geometric patterns provide references for estimating *intrinsic* and *extrinsic camera parameters*. They are mainly employed for calibrating visual (monocular and stereo cameras) and multimodal sensors in various robotics platforms [18, 20], such as surgical robots [21], legged robots [22], industrial manipulators [23], and drones [24]. Markers are also valuable for **multi-sensor synchronization** and fusion, offering a standard reference frame for aligning heterogeneous data streams such as RGB, depth, Inertial Measurement Unit (IMU), and Light Detection And Ranging (LiDAR) [25].

Another primary application domain is **Human–Robot Interaction**, where fiducial markers facilitate reliable communication channels between humans and robotic systems [26]. By placing markers on wearable devices or in shared environments, robots can track human motion, interpret the intent, and support collaborative tasks with robust precision. Fiducial markers are also widely used in **robotic manipulation** [27, 23, 28]

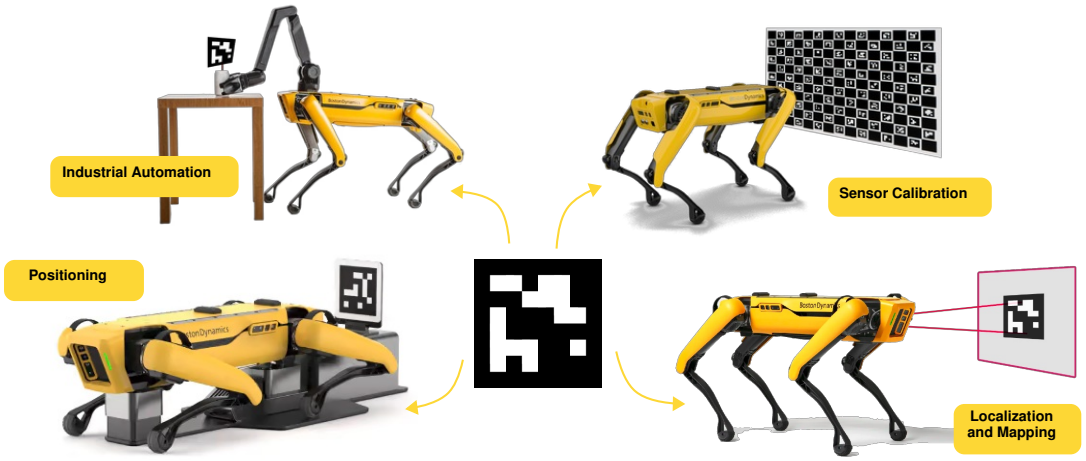


Figure 1.6: Sample fiducial marker applications in robotics.

and **industrial automation** domains to perform tasks requiring high accuracy [29, 30, 31]. Their consistent geometric structure allows robots to reduce reliance on complex perception pipelines, especially in structured environments. Thus, they enable precise placement [32, 33], positioning [34], locating, and alignment in various tasks, such as industrial automation, manufacturing, gripping, and manipulation. In addition, fiducial markers are frequently employed for **object identification**, recognition, and tracking, serving as stable reference points that replace computationally intensive vision algorithms [35, 36]. In such scenarios, using markers can be computationally less expensive than processing an entire visual scene, as the system only needs to identify the marker's pattern and location.

Beyond the applications mentioned above, fiducial markers also play a crucial role in the vision-based **Simultaneous Localization and Mapping (SLAM)** domain [37]. The fundamental goal of SLAM systems is to *construct* a consistent representative map of an unknown environment while simultaneously *tracking* the agent's pose (position and orientation) within it. This creates a challenging circular dependency: an accurate map is required for robot localization, yet accurate localization is needed to build the map [38]. In scenarios where prior maps are unavailable or localization is unreliable, SLAM provides a foundational capability for autonomous operation and has consequently become a central topic in both robotics research and industrial applications [39, 40]. The agent may take various forms, including domestic robots, autonomous vehicles, planetary rovers, and aerial or ground robots [41, 42, 43, 44, 45]. Apart from SLAM variants (such as LiDAR-based SLAM), **Visual SLAM (VSLAM)** is a core approach that relies on visual sensors, including monocular, stereo, or RGB-D cameras [46]. By employing visual data, VSLAM enables cost-effective hardware setups while providing rich geometric and semantic information for scene understanding [47]. A typical VSLAM pipeline is organized into multiple interdependent modules that collectively solve the localization and mapping challenge [48]:

- **Front-end (Visual Odometry)** for raw camera data processing through visual feature extraction, matching, and tracking, and local camera pose estimation [49]. It generally includes a motion estimation pipeline via *PnP* [19] or other methods, selects frames with distinctive visual features as KeyFrames, and continuously performs Local Bundle Adjustment (LBA) (in the back-end) to refine camera poses and 3D points within a small sliding window of processed frames.

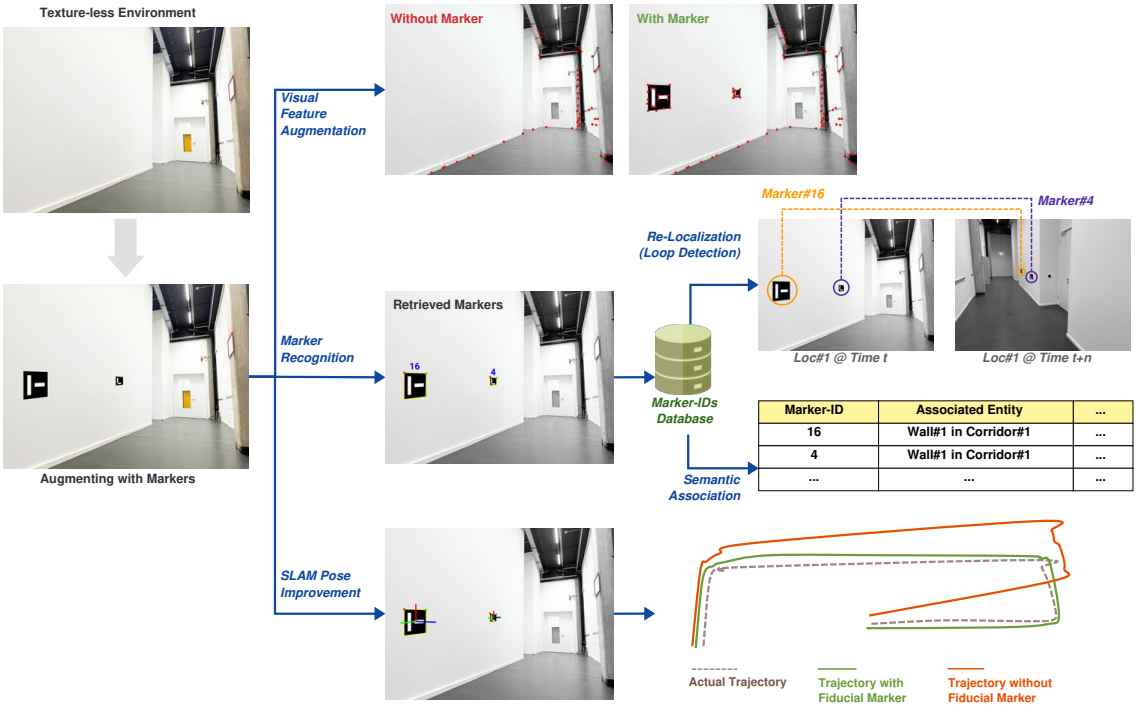


Figure 1.7: The impact of fiducial markers in the Visual SLAM domain: they can aid in highly detectable visual features, stabilizing localization, loop closure detection, and semantic association.

- **Backend (Optimization)** for performing optimization procedures using the front-end data. Modern VSLAM systems typically implement the backend using a **factor graph**, where robot poses and landmarks are the graph's *variable nodes*, and sensor-derived constraints are its *factor nodes*. This structure enables efficient nonlinear optimization of the entire trajectory and map. In addition to LBA, it offers Global Bundle Adjustment (GBA) to optimize the entire map and trajectory when sufficient observations are available.

- **Mapping** to handle the construction and maintenance of the environment map. It decides when to add new KeyFrames, how to maintain map consistency, and how to integrate more observations.

- **Loop Closing** for detecting when the agent returns to a previously visited location (a "loop") and applying global correction to the accumulated drift [50]. It performs GBA and pose graph optimization to correct drift.

In this regard, fiducial markers offer significant advantages by providing fixed, highly reliable, and recognizable *landmarks* for accurate pose estimation and robust mapping [51]. They can be integrated into VSLAM pipelines for several reasons, as shown in Figure 1.7 and discussed below:

- Markers introduce **distinctive** and **highly detectable visual features** into scenes that may be textureless, feature-sparse, or visually degraded. This reduces the likelihood of observing scenes and prevents **tracking failures** (*i.e.,* getting lost) while traversing the environment.

- They contribute to **stabilizing localization** (particularly relative to the initial reference frame), reducing **drift** accumulation over time and improving the overall **global consistency** of the generated map. Each time a marker is observed in the scene, its calculated pose (which is reliable) refines the robot's camera pose estimate from the VSLAM system, thereby preventing the spread of pose drift errors.

- They offer strong cues for **re-localization** and **loop closure detection**. When the robot re-observes a previously seen marker_id, it can instantly recover its position or realize that it has returned to a previously visited location.

- They can be **semantically associated** with specific objects in the environment, augmented with fiducial markers. For example, assigning marker_id = 5 to a wall, marker_id = 8 to a table, and marker_id = 9 to a mirror enables seamless integration of fiducial markers into semantic mapping.

The contribution of these markers to enhancing VSLAM performance, particularly concerning robust initialization and accurate pose estimation, is addressed in detail in the authors' preceding publications [51, 52] and will be further discussed in Section 2.3.

1.3 Literature Review

1.3.1 Classic Fiducial Markers

In general, fiducial markers can be categorized according to various criteria. However, in this thesis, they are classified primarily based on their *geometric shape* as listed below:

I. Non-Square Fiducial Markers

Some fiducial markers are designed with non-square geometries, such as hexagons, circles, or other arbitrary visual patterns. The detection and decoding process for these markers typically relies on identifying a **correspondence point**, which is mainly located at the center of their geometric shape. **Circular-shaped marker libraries**, such as *Intersense* [53], *BullsEye* [54], *Cantag* (circular variant) [55], *CCTag* [56], and *WhyCode* [57], are primarily developed for GPU-based tracking, achieving pixel-level precision by compensating for optical distortions. However, one of the main weaknesses of these markers lies in the detection procedure of correspondence points, which can be unreliable under real-world conditions. Additionally, pose estimation accuracy may degrade due to the absence of a full PnP formulation, and these methods are also more prone to misclassification errors.

Dot-based fiducial marker libraries, such as *Projective-invariant Tag (Pi-Tag)* [58], *RandomDot* [59], and *RUNE-Tag* [60], utilize spatially arranged dots organized into nested layers to enhance detection rates and provide robust occlusion resilience. However, their performance can degrade in noisy environments, where other dot-like features may interfere with markers.

II. Arbitrary Fiducial Markers

Marker libraries with **arbitrary or shape-free designs** offer flexible solutions for 6-Degree of Freedom (DoF) feature tracking and pose estimation. For instance, *Nishino* variants [61, 62] and *d-touch* [63] are based on topological region adjacency features represented as graphs. *Topolo Surface* [64] combines region adjacency with angle data to achieve low false-positive detection rates. Although these approaches provide real-time and accurate marker detection, they are prone to inter-marker dot confusion.

Similarly, *reacTIVision* [65] is an open-source framework designed for real-time detection and tracking of multiple objects, transmitting marker_ids and pose information via a dedicated protocol. Its variety of symbols, such as *amoeba* and *yamaarashi*, enables broad real-time recognition capabilities. However, it lacks robustness-enhancement procedures, such as the Hamming distance [66] or Cyclic Redundancy Check (CRC) [67], making it more sensitive to misidentification. Other systems, such as *Visual Code (Vis-Code)* [68] and *Cybercode* [69], incorporate 2D barcode-like patterns with position-independent coordinates for proper orientation tracking. Since their designs contain numerous small dots, these markers are highly sensitive to noise, leading to detection errors or misclassification in challenging imaging conditions.

Some approaches use highly customizable marker designs, allowing them to be effectively camouflaged within the environment while remaining detectable by the system. *VuMark* [70] is a notable example, primarily used as trackable AR targets, offering millions of unique designs and data-encoding capabilities. *TopoTag* [71] employs topological and geometrical information extracted from the patterns to achieve high robustness and accurate pose estimation. Although these approaches are highly scalable, their detection performance can still be affected by challenging conditions such as extreme viewing angles or partial occlusion.

Finally, STag [72] presents a hybrid solution for stable localization without temporal filtering, using an outer square for homography estimation and an inner circle for refinement, enhancing resilience against pose ambiguity, occlusion, and variable viewing distances. However, its performance can degrade under severe motion blur or partial marker damage, preventing accurate pose estimation.

III. Square Fiducial Markers

The most common variant of fiducial markers used for accurate pose estimation is **square matrix-based**, which uses binary codes and provides four well-defined corner points to enable precise pose estimation. Topology-based fiducial marker approaches are generally more computationally intensive when generating unique marker sets than matrix-based methods. One of the earliest marker libraries, *Matrix* [73], detects patterns in binary images using Connected Component Analysis (CCA). *ChromaTag* [74] introduces a colored fiducial marker with swift detection ($1 - 5$ ms) by transforming images into the L*a*b* color space and segmenting the "a" and "b" channels to read binary codes. Despite robust readout, its performance degrades at long distances or steep viewing angles. The popular open-source library *ARToolKit* [75] employs template matching for marker identification. However, it is sensitive to lighting variations, rotation, and camera angle, often resulting in high re-projection errors and inter-marker confusion. *ARToolKitPlus* [76] addresses some of these issues with automatic global threshold updates and optimized minimum Hamming distance [66].

As a rotation-invariant bi-tonal planar marker system, *ARTag* [77] offers improved robustness under occlusion and intense illumination, primarily due to its edge-based square detection strategy. It is highly resilient to interior pattern overlap and effectively handles false positives through checksum bits. Despite these strengths, *ARTag* suffers from an unoptimized detection pipeline, in which minor inaccuracies in edge detection can propagate widely through the pose estimation process. Inspired by *ARTag*, *CALTag* [78] introduces a self-identifying marker pattern solution to remain robust under occlusion, rotation, uneven illumination, and lens distortion. Its pattern consists of grids populated with uniquely encoded binary patterns, enabling unambiguous identification even in chal-

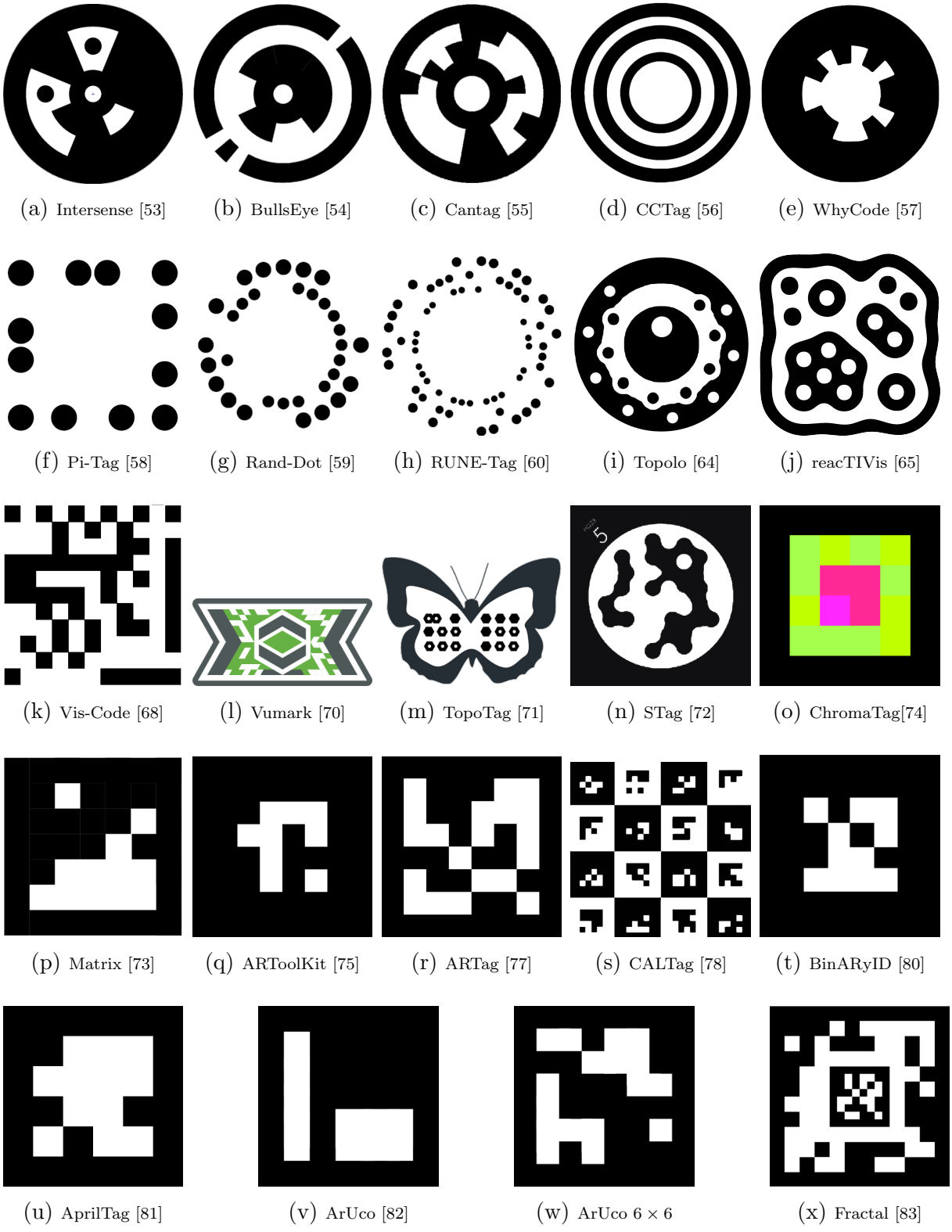
lenging conditions. This enhanced robustness comes at a computational cost, as *CALTag* relies on Random Sample Consensus (RANSAC)-based [79] line-fitting to obtain the approximate coordinates of the occluded parts. *BinARyID* [80] is another approach that employs *bitstrings* and Hamming distance to eliminate rotation ambiguities. However, it remains sensitive to occlusion and partial visibility, and lacks a refinement mechanism for correcting pose estimates.

One of the most widely adopted square fiducial marker systems is *AprilTag* [81], which incorporates maximally decodable binary codes with strong error-correction guarantees. It integrates gradient-based edge detection with reliable line-segment identification, enabling accurate marker recognition even under severe perspective distortion, motion blur, or uneven illumination. Its tag families (*e.g., Tag36h11*, *Tag25h9*) are designed with high Hamming margins, significantly reducing decoding ambiguities and inter-marker confusion. *AprilTag* is also equipped with a fast border-detection pipeline and graph-based image segmentation, resulting in lower bit-error rates and fewer false positives than systems like *ARToolKit*. Similarly, the *ArUco* marker system [82] is another open-source framework widely used in robotics and AR, characterized by a thick black outer border and an inner binary matrix that enables fast and reliable corner detection. The binary matrices are encoded as n-element vectors, where the bit ordering differs across marker dictionaries. For instance, the commonly used 4×4 and 6×6 dictionaries contain 16-bit and 36-bit binary patterns, respectively. *ArUco* also incorporates mechanisms to maximize the inter-marker Hamming distance, reducing misidentification and confusion.

IV. Adaptive and Multi-Modal Fiducial Markers

Beyond the introduced fiducial marker libraries, researchers have also proposed general frameworks to enhance the pose accuracy and robustness of existing marker systems. As a recent system designed to enhance the occlusion and resolution limitations of *ArUco*, *Fractal Markers* [83] employ a nested arrangement of multiple fiducial markers of varying sizes. This hierarchical structure enables reliable detection across a broad range of distances while remaining robust to partial or even severe occlusion, since at least one of the nested layers typically remains visible. The detection pipeline is built upon the *ArUco* framework, extending its marker generation and identification capabilities to support multi-scale patterns. *DeepTag*[1] [84] is another system that employs a Deep Neural Network (DNN) to detect candidate Region of Interests (ROIs) and infer marker identities from clustered keypoint patches. It supports the detection of various existing marker families, such as *AprilTag*, *TopoTag*, and *RUNE-Tag*, but incurs a higher computational cost in exchange for improved recognition. As another sample, *DynaTags* [85] introduced simple paper-based mechanical sliders that dynamically alter marker patterns, enabling adjustable printed fiducial markers. The primary goal is to add interactivity to previously static marker systems while retaining their low cost. Unlike the aforementioned vision-based approaches, *LiDARTag* [86] addresses the challenge of ambient lighting inconsistency by integrating 3D geometric structures with rigidly attached planar fiducial markers. This hybrid design enables reliable detection directly from LiDAR point clouds, allowing the system to operate even in complete darkness. However, *LiDARTag* is not intended for conventional camera-based workflows.

[1] `https://herohuyongtao.github.io/research/publications/deep-tag/`



(a) Intersense [53] (b) BullsEye [54] (c) Cantag [55] (d) CCTag [56] (e) WhyCode [57]

(f) Pi-Tag [58] (g) Rand-Dot [59] (h) RUNE-Tag [60] (i) Topolo [64] (j) reacTIVis [65]

(k) Vis-Code [68] (l) Vumark [70] (m) TopoTag [71] (n) STag [72] (o) ChromaTag[74]

(p) Matrix [73] (q) ARToolKit [75] (r) ARTag [77] (s) CALTag [78] (t) BinARyID [80]

(u) AprilTag [81] (v) ArUco [82] (w) ArUco 6 × 6 (x) Fractal [83]

Figure 1.8: Some instances of classic fiducial marker libraries used in computer vision, robotics, and augmented reality domains.

V. Discussions on Marker Variants

Figure 1.5 depicts several fiducial marker libraries introduced in this section, which are available for research and commercial purposes. Despite their visual differences, all li-

braries aim to enhance tasks such as long-range detection, pose accuracy, or visual distinctiveness. These printable markers are typically placed in real environments to embed machine-readable information onto surfaces.

Considering the diversity of methods and their respective pros and cons, selecting an appropriate marker library can directly impact the achievable precision and robustness of pose estimation. In practice, the performance of any fiducial marker system is heavily influenced by environmental and system factors. For instance, insufficient illumination, glare, or inconsistent exposure may disrupt thresholding, corner detection, and edge extraction procedures of the fiducial marker recognition pipeline. Motion blur and optical de-focus caused by rapid camera movement can also distort the marker's geometry, reducing decoding accuracy and reliability. Viewing angle and distance are other vital factors. While extreme perspectives introduce severe projective distortion, long-range observations reduce pixel density, thereby hindering feature detection. Additionally, camera calibration errors, lens distortion, inaccurate marker-related configuration, and surface reflections can negatively affect pose estimation. Thus, achieving stable and accurate marker-based localization requires a robust library and appropriate environmental conditions.

1.3.2 Blended and Unobtrusive Fiducial Markers

In addition to conventional paper-based fiducial markers, several research efforts have proposed *blended* landmarks that can be fused into the environment, making them less visible to humans while remaining detectable by dedicated perception systems. These approaches typically rely on specialized *sensing modalities* or *optical filtering*, as the markers are not designed to be captured reliably by standard RGB cameras. For instance, MechaTag [87] introduces a mechanically engineered fiducial marker system in which patterns are formed by *through-holes* and *blind-holes* embedded in rigid surfaces composed of reflective and non-reflective materials. The marker information is encoded in the geometric and topological properties of these features, and a detection algorithm exploits the distinct hole configurations. The primary objective of MechaTag is to minimize recognition errors and improve repeatability, particularly under low-illumination and visually challenging boundary conditions, where traditional planar markers tend to fail. However, the approach is limited to rigid surfaces and requires physical modification of the host object to encode information. SeedMarkers [88] introduce shape-independent topological fiducial markers designed with an emphasis on aesthetic integration into physical objects. They are embedded into arbitrarily shaped surfaces and can be fabricated using laser cutting or 3D printing, often directly derived from technical drawings during the design process. Despite their flexibility, the absence of an explicit error-correction mechanism makes SeedMarkers vulnerable to occlusion and partial visibility. Furthermore, accurate 6-DoF pose estimation requires prior knowledge of each marker's exact geometry, which limits scalability and complicates deployment in unstructured environments. In another work, the authors introduced A4LidarTag [89]: fiducial marker patterns composed of circular holes specifically designed for LiDAR–camera systems, targeting applications such as extrinsic calibration and robotic perception. The marker encodes spatial information through depth discontinuities introduced by the holes, which are reliably captured by LiDAR measurements. While it offers advantages, the framework is optimized only for close-range operations and cannot be used for camera-only perception. ACMarker [90] is another fiducial marker system detectable by acoustic camera-based sensors for underwater robotics applications. It primarily addresses the challenges posed by blended landmarks in water turbidity and illumination conditions. However, the solution suffers

from high computational costs and very limited application in the robotics domain.

Beyond blended markers, several approaches aim to integrate less obtrusive fiducial markers into environments in order to reduce visual clutter while preserving reliable detectability. Transparent Random Dot Markers (TRDM) [91] is one such system that employs transparent substrates with sparsely and randomly distributed printed dots, rendering the markers nearly imperceptible under normal viewing conditions. However, reliable detection remains challenging, as geometric point matching requires establishing consistent pairwise relationships among detected dots, which can be sensitive to noise and partial occlusion. BrightMarkers [92][2] target invisible object tagging by embedding 3D-printed fluorescent filaments directly within objects. Their detection success relies on Near-Infrared (NIR) cameras combined with appropriate illumination (the optimal wavelength of 763nm and the emitted fluorescence peak of 775nm), enabling marker-based tracking without altering the object's appearance or geometry. BrightMarker supports integration with objects of varying colors and enables reliable localization using a light source and a camera filter tuned to the material's fluorescence characteristics. Nevertheless, the marker fabrication pipeline (particularly the fluorescent dye injection step) is highly sensitive and can directly affect detection performance.

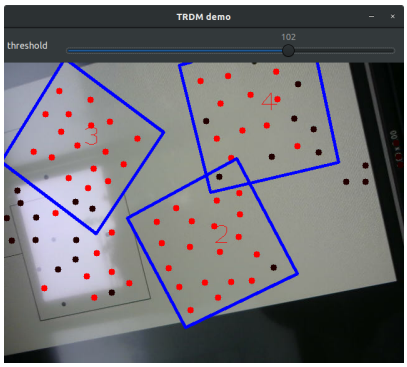
InfraredTag [93] is another discreet and durable fiducial marker that can be printed or embedded in physical objects, enabling invisible tagging and rapid detection via IR sensing. However, their fabrication is inherently constrained to the IR spectrum, limiting material choices and production flexibility. Furthermore, their thin and embedded design makes manual localization difficult, complicating the alignment of IR torches or cameras during deployment. As a similar approach, InfraStructs [94] embed information directly within physical objects for imaging in the Terahertz (THz) spectrum, making them particularly suitable for digitally fabricated artifacts. However, THz imaging systems remain relatively immature and costly, limiting their practical deployment in robotics. Moreover, such systems typically rely on specular surface reflections to obtain a measurable signal, which constrains robustness under varying surface materials, orientations, and environmental conditions. Aircode [95] embeds fiducial markers during object fabrication, enabling seamless, visually unobtrusive integration for applications such as robotic grasping. The marker is formed by a set of carefully designed air pockets located beneath the object's surface and can be embedded under flat or gently curved geometries. Despite these advantages, AirCode suffers from a slow decoding process that can take tens of seconds (depending on the viewpoint), limiting its applicability for real-time robotics.

Fig. 1.9 depicts some of the introduced unobtrusive fiducial marker systems. Considering the aforementioned research, **transparency** and **fabrication-level integration and flexibility** emerge as primary objectives in the design of unobtrusive fiducial markers. While these approaches successfully reduce visual impact and preserve object aesthetics, they often introduce trade-offs in *material choice*, *design flexibility*, and *supported encoding schemes* for reliable recognition and information retrieval. In addition, practical limitations remain in terms of *fabrication cost* and complexity, *sensor compatibility*, and the robustness of the associated *detection algorithms*, revealing open gaps that motivate further investigation, including the topic of this thesis.

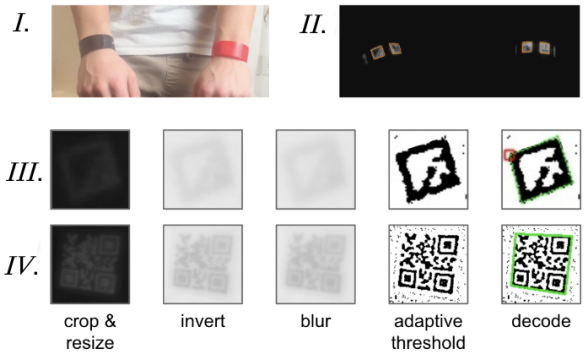
1.3.3 Visual SLAM Systems

Visual SLAM (VSLAM) systems have matured over the past few years, and several frameworks have played a vital role in this development. As a fundamental classifica-

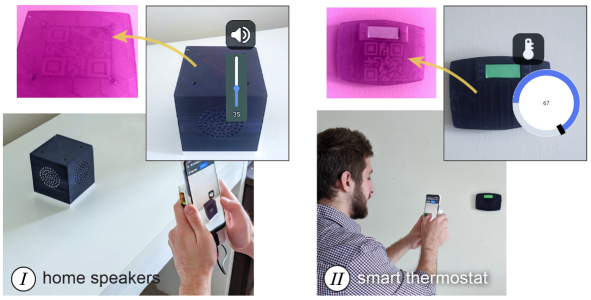
[2] `https://hcie.csail.mit.edu/research/brightmarker/brightmarker.html`



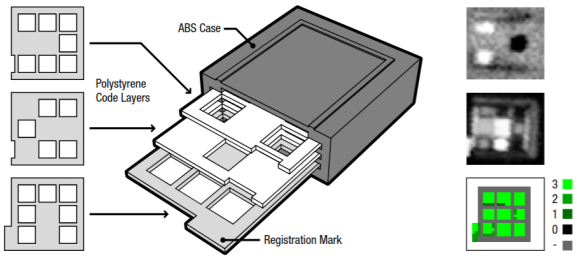
(a) TRDM [91] detection software revealing the marker patterns.



(b) BrightMarker's [92] image processing pipeline, from object tracking to pattern decoding.



(c) InfraredTags [93] sample applications in (I) home speaker volume and (II) thermostat temperature adjustment.



(d) The structure of InfraStructs [94] with three layers of encoded data.



(e) AirCode [95] tagging tool, from input mesh to user-specified region and embedded data.

Figure 1.9: Concepts of unobtrusive and blended fiducial markers for computer vision, robotics, and augmented reality domains: **(a)** AirCode [95], **(b)** InfraStructs [94], and **(c)** InfraredTags [93].

tion, VSLAM methods can be divided into **indirect** (feature-based), **direct**, and **hybrid** approaches [96]. *Indirect methods* rely on detecting and matching salient feature points across frames, offering robustness to photometric variations at the cost of additional computational overhead. In contrast, *direct methods* estimate camera motion by directly minimizing photometric error with respect to pixel intensities, enabling denser scene reconstruction but increasing sensitivity to illumination changes and the challenges of large-scale optimization [97]. To balance these trade-offs, *hybrid approaches* combine elements of both paradigms, typically using feature-based methods for initialization or correction, while employing direct optimization to improve accuracy.

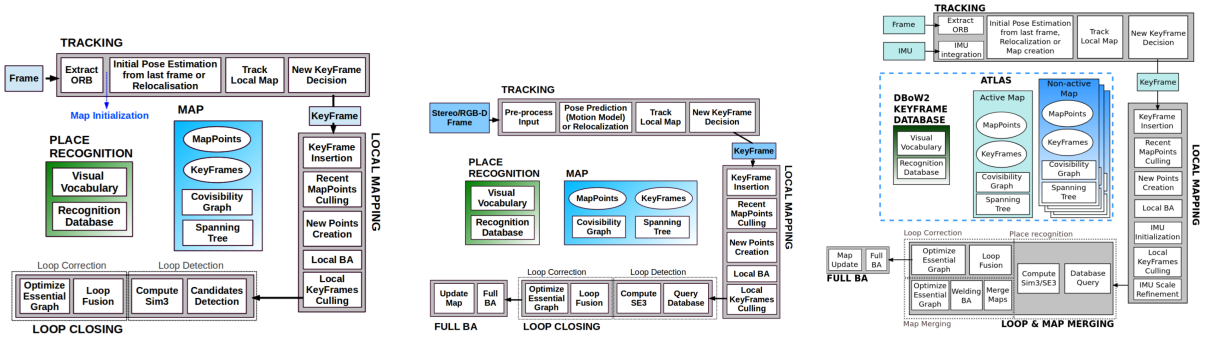
MonoSLAM, introduced by Davison *et al.* in 2007 [98], was the first real-time monocular Visual SLAM (VSLAM) system, using an Extended Kalman Filter (EKF)-based indirect framework to jointly estimate camera motion and sparse $3D$ landmarks. While it

lacked global optimization and loop-closure capabilities, MonoSLAM laid the foundation for VSLAM, producing maps based on sparse landmark representations. Klein *et al.* proposed PTAM [99], a parallel tracking–mapping architecture that estimates camera pose in real time and performs joint optimization via Bundle Adjustment (BA), significantly improving performance over MonoSLAM. DTAM [100] was a direct VSLAM approach that used dense tracking and mapping by aligning full-image frames to estimate camera motion and scene depth, producing detailed reconstructions at the expense of high computational cost. Building on or inspired by these baselines, various VSLAM frameworks have been developed, each addressing different gaps.

A research topic focuses on **multi-modality** and **sensor fusion** to improve environmental understanding. Methods such as CoSLAM [101] and MultiCol-SLAM [102] employ multiple cameras to improve tracking and map reconstruction accuracy. While such approaches enhance performance in occlusion, texture sparsity, and dynamic scenes, they share common drawbacks, including increased hardware complexity, higher computational cost, and limited scalability. Other works, like CamVox [103], VIRAL-SLAM [104], and Ultimate SLAM [105], fuse visual data with complementary sensing modalities, such as LiDAR and IMU. While these approaches significantly improve long-range accuracy and performance under challenging conditions, they face challenges related to sensor synchronization, system complexity, and computational overhead arising from tightly coupled multi-threaded processing.

Pose estimation accuracy is another research line that relies on robustly (re)detecting patterns, features, or external landmarks. Several point- [109, 110] and line-based [111, 112] VSLAM systems have been proposed to improve performance in low-texture or featureless environments, While they enhance trajectory accuracy, they require complex feature tracking, geometric primitives, or parameter tuning to maintain reliable performance. Other works integrate high-level [113, 114, 115, 116] or deep visual features [117, 118, 119, 120, 121, 122, 123, 124, 125, 126] for accurate feature extraction and matching. However, they often fail to operate in real time and require complex backbones or preprocessing pipelines. Among these approaches, ORB-SLAM variants [106, 107, 108] (shown in Fig. 1.10) serve as highly accurate baselines and have been widely extended due to their support for diverse vision sensors and reliable performance. Furthermore, some works integrate external visual landmarks (primarily fiducial markers) to aid in tracking and pose estimation, as discussed in **§1.3.4**.

Several works focus on the applicability of VSLAM approaches in **dynamic environ-**



(a) ORB-SLAM [106]. (b) ORB-SLAM 2.0 [107]. (c) ORB-SLAM 3.0 [108].

Figure 1.10: The architecture of ORB-SLAM variants that have served as the foundation for numerous VSLAM frameworks.

ments by using feature filtering to handle moving objects. In this regard, frameworks like DS-SLAM [127], SOF-SLAM [128], DynaSLAM [129, 130], DynamicSLAM [131, 132], SaD-SLAM [133], and others [134, 135] integrate semantic segmentation and object detection pipelines into VSLAM. Semantic VSLAM methods, such as [136, 137], estimate geometric maps and incorporate semantic objects in the environment to jointly optimize the robot pose and object landmarks. Other approaches aim to construct dense, detailed environment representations using surfel-based mapping [138], Gaussian scene representations [139, 140, 141], or neural implicit models [142], thereby enabling high-fidelity 3D reconstruction and novel view synthesis. While these methods improve localization and mapping accuracy in dynamic scenes, they share limitations in computational cost, reliance on segmentation quality, and sensitivity to feature sparsity.

Several **resource-constrained** VSLAM systems target low-computation devices or leverage offloading, such as EdgeSLAM [143, 144], ProSLAM [145], and VPS-SLAM [146]. These approaches enable real-time operation with limited hardware while maintaining mapping and localization performance. However, they often result in reduced functionality, limited support for features or objects, and increased system complexity due to offloading or simplification strategies. Other works target **versatility** in design, prioritizing adaptability, ease of development, and multi-sensor support. Frameworks like OpenVSLAM [147], OV^{2}SLAM [148], and DROID-SLAM [149] enhance flexibility, accuracy, and cross-platform usability in the VSLAM domain. Such benefits generally come with limited hardware and preprocessing requirements in dynamic or constrained environments.

In summary, the reviewed VSLAM frameworks cover a wide spectrum of objectives and target categories. Across them, primary directions and trends, such as deep learning-based scene understanding and the use of external landmarks to improve accuracy, are more detailed in §2.2 and **Paper#2** (§5).

1.3.4 Marker-based Visual SLAM

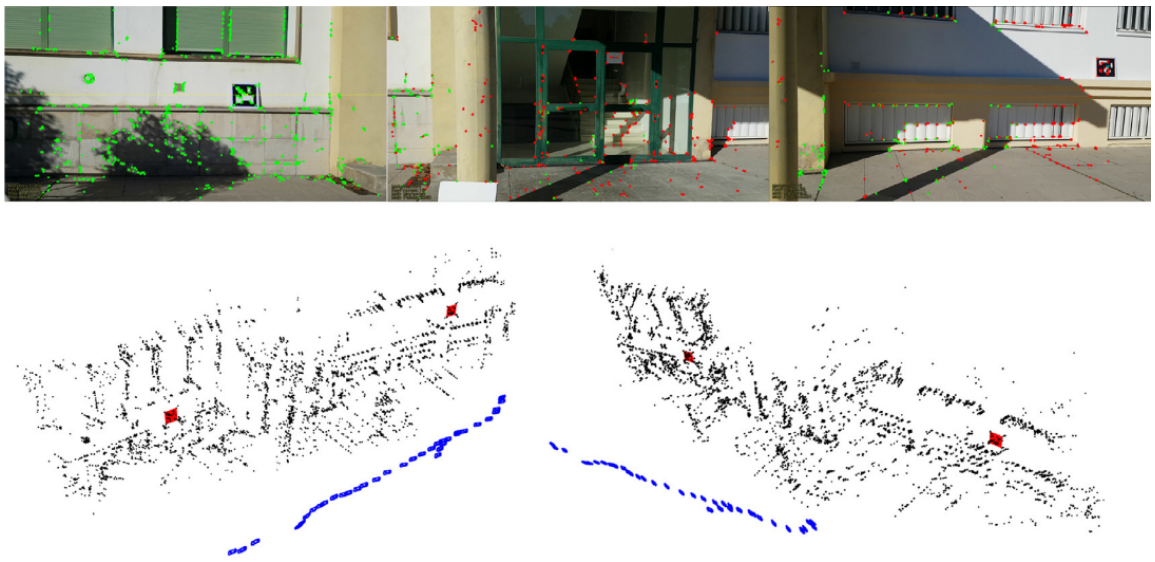
Some VSLAM approaches incorporate artificial landmarks, primarily fiducial markers, to enhance scene understanding and improve the robustness and accuracy of camera pose estimation. Marker-based VSLAM systems benefit from reliable data association, reduced drift, and improved performance in low-texture, repetitive, or dynamic environments, where purely feature-based methods often struggle.

In this regard, Muñoz-Salinas and Medina-Carnicer introduced UcoSLAM [150], a hybrid VSLAM framework that combines natural features with artificial fiducial markers to improve robustness and automatically recover the metric scale of the environment. The method addresses the instability, repetitiveness, and poor trackability of natural landmarks by supporting *keypoints-only*, *marker-only*, and *mixed* operating modes, enabling reliable operation even in feature-sparse or marker-free scenes. UcoSLAM performs tracking through map correspondence search, re-projection error minimization, and re-localization upon failure, and includes a marker-based loop closure mechanism while remaining agnostic to the choice of feature descriptor. Despite these advantages, its multi-threaded architecture can lead to increased computational overhead in time-critical scenarios. TagSLAM [152] is another framework that uses AprilTags for localization and mapping. It operates exclusively in a marker-only mode, requiring continuous visibility of fiducial markers for reliable tracking and pose estimation. However, the system is not a full Visual SLAM framework and primarily relies on markers to improve trajectory estimation. In another work [153], authors propose a fiducial marker-assisted

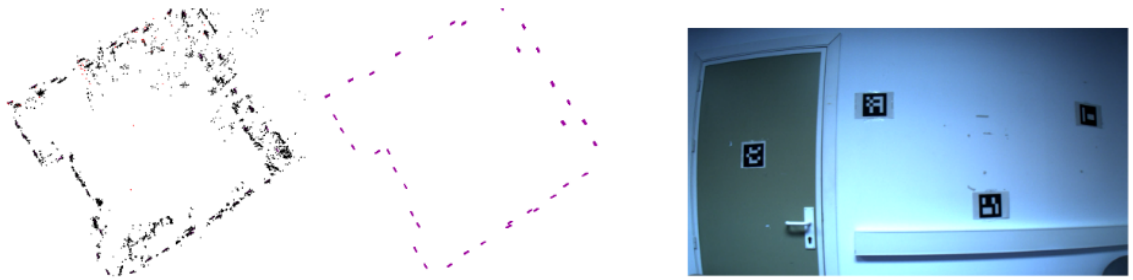
initialization method that jointly employs marker points and ORB features to estimate a scale-consistent visual trajectory. While showing improvements in speed and accuracy, the system remains limited to the initialization stage rather than incorporating markers for the SLAM pipeline.

sSLAM [151] proposes a novel approach that substantially reduces the computational time and memory requirements without noticeably degrading tracking accuracy. The system builds an initial map using both natural keypoints and artificial markers, and upon completion, it discards the keypoints and keeps only the markers as long-term, stable landmarks. While it enhances robustness at runtime, it relies heavily on reliable, sufficiently visible marker detection. Similarly, Wang *et al.* [154] extend SPM-SLAM by improving initialization, keyframe insertion, and relocalization, achieving higher robustness and accuracy under rotational motion, while addressing planar-marker pose ambiguity through a novel multi-image marker pose estimation strategy.

Some of the introduced approaches are shown in Fig. 1.11. The mentioned approaches primarily emphasize geometric map construction and marker-based loop closure, while largely overlooking the potential of fiducial markers as carriers of semantic information. Consequently, the use of markers for higher-level semantic understanding and scene interpretation remains an open research gap.



(a) Map reconstruction using UcoSLAM [150].



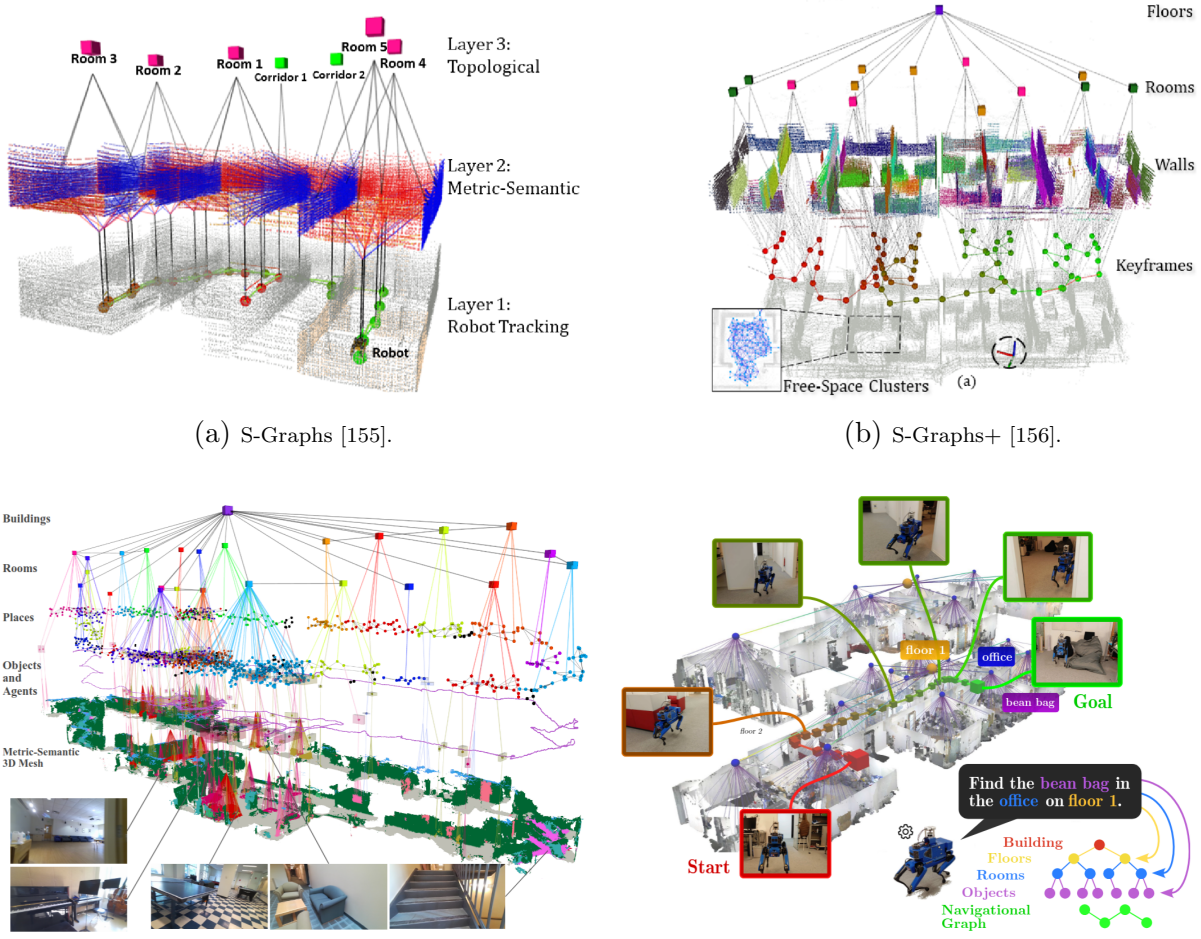
(b) Map reconstruction using sSLAM [151].

Figure 1.11: Samples of marker-based Visual SLAM frameworks.

1.3.5 High-Level Scene Representation

While the reconstructed maps of the environments provide accurate *geometric layouts*, they remain limited in their ability to capture **semantic structure** and **relational information**, motivating the adoption of higher-level representations, such as **3D scene graphs**. In particular, 3D scene graphs represent the environment as a graph where *nodes* represent scene entities such as objects, surfaces, or spaces, and *edges* encode spatial, functional, or semantic relationships between them (*e.g.,* adjacency or containment) [159]. This abstraction enables compact scene descriptions, supports higher-level reasoning, and facilitates next-stage tasks, such as decision-making and navigation in a reconstructed map. In this context, 3D representations and scene graphs have emerged as powerful tools to describe scenes by organizing objects, spatial relationships, and semantic attributes within a unified, interpretable framework [160]. Fig. 1.12 depicts some of the solid baselines for generating 3D Scene Graphs.

Apart from works like [159, 160, 159, 161] that generate hierarchical representations of environments by connecting detected entities, some research aims to combine them with SLAM-reconstructed maps to enhance environmental understanding. In this regard, S-Graphs [155] is a LiDAR-based framework that tightly couples SLAM with 3D scene graphs into an optimizable factor graph, generating multi-layered hierarchical environment maps. It provides comprehensive robotic situational awareness by combining robot



(a) S-Graphs [155].

(b) S-Graphs+ [156].

(c) Hydra [157].

(d) HOV-SG [158].

Figure 1.12: Some of the solid research works for making 3D Scene Graphs.

poses, geometric maps, and hierarchically organized semantic entities, along with their topological relationships, into a single graph, while improving SLAM-level localization and mapping performance by exploiting semantic information. Its extension, *S-Graphs+* [156], further unifies the pose graph and a four-layer hierarchical 3D scene graph (containing keyframes, walls, rooms, and floors) within a single real-time optimizable factor graph. By jointly estimating robot poses and structured semantic maps using novel room and floor segmentation from wall planes and free-space, *S-Graphs+* significantly improves localization accuracy while enriching high-level situational awareness. More recently, S-Graphs 2.0 [162] extends this formulation by using the hierarchical structure of indoor environments to enable efficient data management and optimization. By introducing floor-aware semantics, floor-based loop closure, and multi-level optimizations, it achieves state-of-the-art accuracy while reducing computational cost and enabling hierarchical mapping up to an order of magnitude faster in large multi-floor environments. It should be noted that, apart from their robust map reconstruction and representation, S-Graphs variants [155, 162, 156] are limited to the exclusive use of LiDAR data.

Since vision-driven maps are typically sparser and noisier than LiDAR-based reconstructions, building accurate and consistent 3D scene graphs from purely visual inputs remains particularly challenging. In this context, Hydra [157] introduces a real-time spatial perception system that incrementally constructs multi-layer 3D scene graphs from local geometry to places and rooms. It supports hierarchical loop closure detection and global graph optimization, enabling online scene graph reconstruction with accuracy comparable to offline methods. However, it does not cover a full SLAM system and treats SLAM and 3D scene graph generation as separate pipelines. HOV-SG [158] is another vision-based approach that employs open-vocabulary vision foundation models to construct hierarchical 3D scene graphs with room- and floor-level semantics, enabling multi-floor navigation and long-horizon planning via language-model-driven queries. Despite its robust environment modeling, it primarily represents spatial hierarchies rather than estimating camera trajectories, and requires ground-truth semantics and an entire-environment map to construct the scene graph offline. Considering the mentioned trade-offs, a clear gap and an untapped direction exist in the community.

1.4 Research Motivation & Objectives

1.4.1 Research Motivation

With the rapid pace of technological advances, the presence of *intelligent agents*, including robots, autonomous vehicles, and smart devices, has become increasingly widespread across numerous application domains. In everyday life, robotic vacuum cleaners (commonly known as *RoboVac* or *Roomba*) have become standard household assistants, while newer generations of domestic service robots are being developed to perform more complex household chores. In industrial environments, robots used in automotive manufacturing, warehouse logistics, and large-scale storage facilities are becoming increasingly autonomous and capable of executing tasks with minimal human intervention. However, for these systems to operate safely and effectively, they must perceive and interpret complex, dynamic real-world environments. This often requires computationally intensive **perception** pipelines, high-resolution sensors, and powerful onboard hardware.

As a response, **enriching environments** with objects that embed machine-readable information can substantially simplify perception, decision-making, and task execution for intelligent agents. Such augmentation unlocks a wide range of capabilities, includ-



Figure 1.13: An environment augmented with several fiducial markers placed on dissimilar objects and entities. The illustration is an AI-generated image that the author modified to include additional fiducial markers.

ing faster and more reliable authentication, improved operational safety, and a more structured understanding of the environment. As robotic systems become increasingly complex, the strategic use of information-rich **fiducial markers** offers a practical means of enhancing robustness, interpretability, and performance. Since each marker encodes a unique identity, objects can be explicitly labeled with dedicated patterns, like assigning one printed marker to a *window* and another to a *desk*. For example, extracting information from printed markers is far more computationally efficient than relying on full-scene interpretation by DNNs, which typically requires substantial computational power and costly hardware. Thus, robots can significantly reduce the search space during perception, thereby enabling **semantic understanding** at low computational cost. Additionally, marker-augmented objects provide precise **geometric cues**, including the object's pose, distance, and orientation relative to the robot.

Although fiducial markers offer substantial benefits for robotic perception, a predictable challenge arises when imagining future environments populated by various autonomous robots. If every object in an environment is labeled with visually prominent fiducial markers to support robotic understanding, the surroundings would quickly become **aesthetically unappealing**, **intrusive**, and **overly cluttered** for human occupants. For instance, the scene illustrated in Figure 1.13 is an office space filled with fiducial markers attached to various objects to convey their semantic identities for lightweight robotic perception. It can be seen that while markers are helpful for robots, working in an environment saturated with them would be distracting and visually disruptive for humans, particularly since the markers provide no functional value to humans.

This conflict between *robot-friendly visibility* and *human-friendly design* of fiducial markers in robotics is a central motivation for this thesis. To support future robotic systems that operate seamlessly around people, there is a growing demand for landmarks

that retain the functionality of fiducial markers while minimizing visual impact. Accordingly, this thesis aims to explore and develop fiducial markers that possess unobtrusive, non-distracting, and ideally invisible characteristics, explicitly tailored for robot-centric applications. Such markers **blend naturally** into surroundings, preserving the **visual integrity** of indoor and outdoor spaces, while remaining **reliably detectable** by robots.

Considering the characteristics of CSRs introduced in Section 1.2.2, these materials present a promising potential for designing the demanded fiducial markers. In this thesis, we refer to such CSR-based fiducial markers as **iMarkers**. The core idea is that, by replacing traditional printer toner or ink with microscopic Cholesteric Spherical Reflector shells, it becomes possible to fabricate markers that remain visually unobtrusive (or entirely invisible) to humans while still being reliably detectable by robotic sensors. Since CSRs can exhibit strong reflection bands outside the visible spectrum, they can allow information to be embedded on surfaces without altering their human-facing appearance. This enables augmenting environments with markers that preserve aesthetic integrity while still providing robots with robust landmarks for perception, localization, and interaction.

1.4.2 Research Objectives

Thus, the primary objective of this thesis is to investigate the design and use of **iMarkers** and to evaluate whether robots can rely on them to achieve enhanced robot understanding, primarily defined as **situational awareness** [163]. In this context, situational awareness refers to the robot's ability to *perceive*, *interpret*, and *interact* with its operating environment through recognizing objects, understanding spatial layouts, and tracking its own pose and motion. This thesis lays the foundation for employing iMarkers in robotics, covering their detection through hardware and software and demonstrating their integration into robotic tasks as a proof of concept. Accordingly, the objectives of the thesis are structured around the following research questions:

- **RQ1.** What are the **characteristics of iMarkers**, and which **hardware and software strategies** enable their reliable detection and sensing in robotic environments? (§2.1)
- **RQ2.** How do **state-of-the-art Visual SLAM approaches** exploit environmental structure, and what **emerging trends and limitations** affect their ability to support situational awareness? (§2.2)
- **RQ3.** Can **integrating artificial landmarks** improve a robot's understanding of its surroundings and enhance situational awareness? (§2.3)
- **RQ4.** Can **iMarkers** compete with conventional fiducial markers without compromising robotic **task performance**? (§2.4)
- **RQ5.** Can robots achieve **robust situational awareness** without artificial landmarks, while using them only as **supplementary aids** to enhance perception and environmental understanding? (§2.5)

1.5 Significance of the Study

This thesis presents the first comprehensive exploration of **iMarkers from a robotics perspective**, covering three core elements: **(1)** the design and evaluation of dedicated

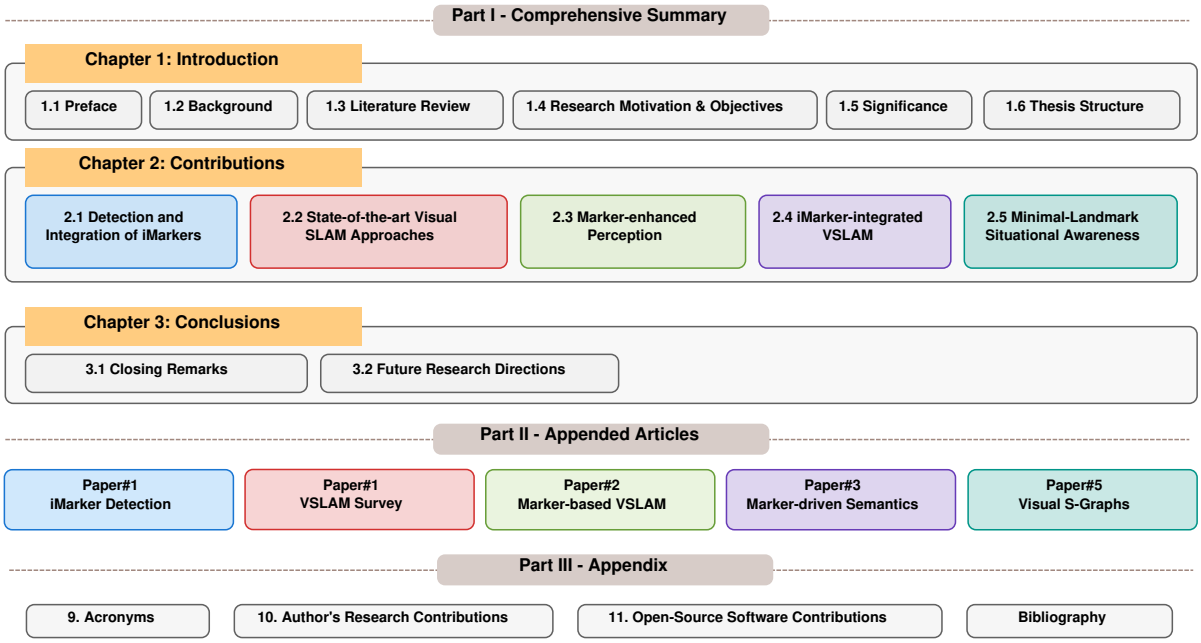


Figure 1.14: The overall structure of the thesis.

sensing hardware, **(2)** the development of robust **detection and recognition algorithms**, and **(3)** their integration into **robotic pipelines** for enhancing situational awareness and environment understanding. As iMarkers are novel solutions emerging from the TRANSCEND team's interdisciplinary collaboration, establishing the foundations for their practical use in robotics was a fundamental challenge. It should be emphasized that the **material properties** and **fabrication procedures** of these markers have been previously introduced in the PhD thesis of **Dr. Deniz Işınsu Avşar**[3].

In this regard, the significance of this study is threefold:

- **Foundational Integration of iMarkers in Robotics**: Establishing the first systematic exploration of iMarkers as *robot-perceivable yet human-inconspicuous* landmarks, and characterizing their essential properties for practical use in robotic systems.
- **End-to-End Detection Sensors and Pipelines**: Developing a complete detection stack, from custom-designed sensing hardware to robust software algorithms, thus providing the first functional pipeline capable of reliably detecting and decoding iMarkers in real-world conditions.
- **Robotic Frameworks for Situational Awareness**: Designing and evaluating a series of robotic perception and VSLAM frameworks, from **fully marker-based** pipelines to **hybrid approaches** leveraging iMarkers, and finally to **marker-optional** systems where artificial landmarks act as auxiliary cues to enrich semantic and spatial understanding.

1.6 Thesis Structure

As also shown in Figure 1.14, the content of the thesis is categorized as follows: Chapter 2 presents the overall contributions of this thesis and their alignment with the appended

[3] `https://www.uni.lu/fstm-en/people/deniz-isinsu-avsar/`

papers in Part II. It begins by introducing how iMarkers can be integrated into robotic systems and by describing their detection using custom-designed sensors and software pipelines (§2.1). In §2.2, state-of-the-art VSLAM approaches, their design choices, emerging trends, and open challenges are analyzed and systematically categorized into coherent classes. Next, §2.3 outlines a marker-based VSLAM pipeline as a proof-of-concept, describing how conventional printed markers can enhance environmental understanding and scene modeling. In §2.4, we extend these ideas by examining how iMarker-based detection pipelines can support (or even replace) standard sensing modalities for environment understanding. Finally, §2.5 introduces a comprehensive framework for vision-based situational awareness that operates independently of markers, but can incorporate printed and iMarker landmarks as optional semantic enrichment entities.

Chapter 3 provides the concluding remarks of this thesis (§3.3), summarizing the core findings and reflecting on the overall contributions of the work. It also outlines promising future research directions and discusses the current limitations encountered throughout the study (§3.2), offering guidance for continued advancement in iMarker technology, detection pipelines, and robotics-oriented situational awareness frameworks.

Part II presents the five primary papers contributed during the author's PhD. The papers are organized by the *thematic structure* used to summarize the thesis contributions rather than the *publication date*. Accordingly, **Paper#1** (§4) details the iMarker sensing and detection pipeline. **Paper#2** (§5) provides a broad study of VSLAM systems. **Paper#3** (§6) corresponds to the marker-based Visual SLAM approach. **Paper#4** (§7) extends perception to include iMarker detection. **Paper#5** (§8) relates to the situational-awareness framework.

Finally, Part III includes the list of acronyms (§9), all publications produced during the Ph.D. career (§10), the open-source contributions (§11), and the complete bibliography.

2

Contributions

The research contributions of this thesis can be traced through two complementary aspects: The first aspect consists of **peer-reviewed research publications** produced by the author during the doctoral studies. Out of a total of **fifteen peer-reviewed publications**, five directly contribute to the scientific content of this thesis, while the remaining publications represent supplementary research contributions. These five contributions are introduced in the following sections, each addressing a specific **Research Question (RQ)** of the thesis defined in §1.4.2. The second aspect comprises **open-source software contributions** developed by the author to support verification, reproducibility, and further research. Both aspects are listed in **Part III (Appendices)** under the sections entitled "Author's Research Contributions (§10)" and "Open-Source Software Contributions (§11)," respectively.

2.1 Detection and Integration of iMarkers (RQ1)

2.1.1 Overview

Considering the interdisciplinary collaboration within the TRANSCEND project, the **material science aspects** of iMarker fabrication were extensively studied by **Dr. Deniz Işınsu Avşar**[1]. This includes the experimental development of CSR formulations, validation of component stability, and optimization of the fabrication workflow. However, the **robotics-oriented considerations** essential for adapting iMarkers to practical robotic use were led and contributed by the author.

It is important to note that iMarkers are not the first fiducial marker solutions that aim to blend markers into their surroundings seamlessly. However, unlike previous approaches, iMarkers uniquely offer **transparency**, **flexibility** (in pattern coding and light spectrum), and **unobtrusiveness** thanks to the optical properties of CSRs [12]. These distinguishing characteristics and their implications for robotic perception are thoroughly detailed in **Paper#1** (§4), directly addressing the concerns of **RQ1** (§1.4.2). Accordingly, iMarkers are composed of patterned regions containing microscopic CSRs, while the remaining areas are filled with transparent UV-curable glue to preserve the marker's

[1] `https://www.uni.lu/fstm-en/people/deniz-isinsu-avsar/`

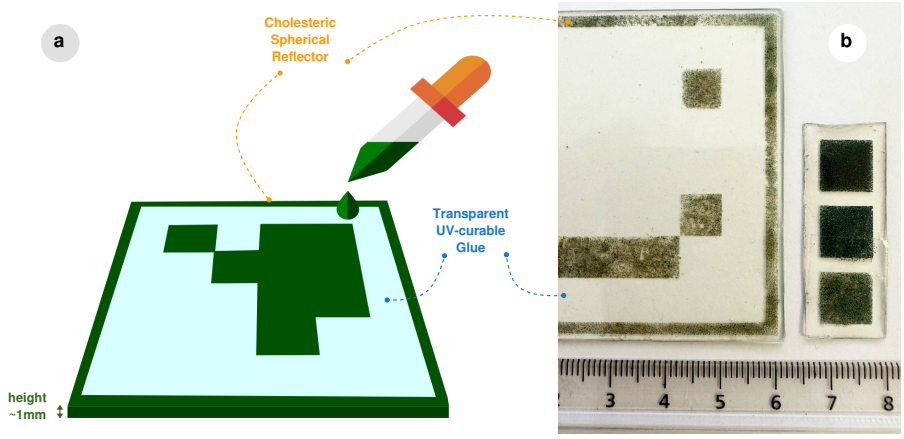


Figure 2.1: CSR shells in the structure of iMarkers (taken from [164]): **(a)** an iMarker featuring ArUco patterns filled with CSRs, **(b)** an iMarker displayed next to visible-range CSR particles, illustrating various shades of green for fabrication.

shape, as illustrated in Figure 2.1. A second fabrication method (which was still under investigation during the project) relies on spraying CSR shells directly onto surfaces to form marker patterns [164].

In this regard, iMarkers offer flexible fabrication options and can be structured using any desired coding pattern for robust recognition and information encoding. Given the wide range of fiducial marker libraries introduced in §1.2.3, selecting an appropriate marker variant requires satisfying several key criteria:

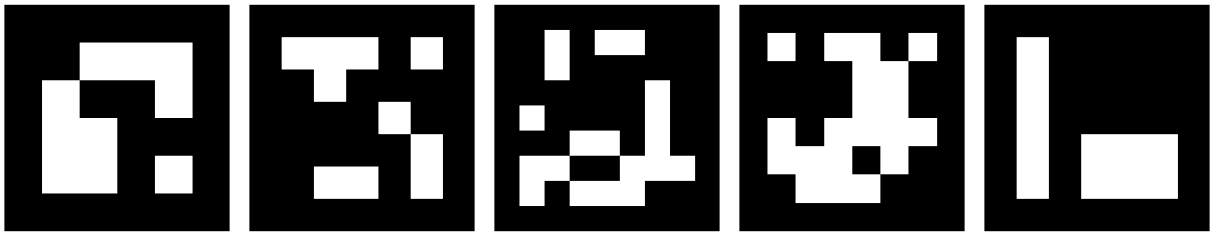
- High detection **reliability** with a low false-positive rate,
- **Ease of fabrication**, ensuring that the marker pattern can be reliably reproduced using CSR shells, given the early-stage constraints of the fabrication pipeline; and
- Suitable for **short-range indoor applications**, serving as an initial proof of concept for the potential of iMarkers.

After evaluating several alternatives, the ArUco family [82] was selected due to its balance of detection robustness, simplicity, and fabrication feasibility. In particular, the original ArUco dictionary was preferred over more intricate designs because it is significantly easier to fabricate using cholesteric CSRs. Accordingly, Figure 2.2 illustrates this rationale: among various patterns, the white regions of ArUco markers should correspond to CSR-filled areas, while the black regions are formed using UV-curable glue. It is visually evident that the original ArUco marker pattern is significantly easier to fabricate as an iMarker (given the outlined criteria), which ultimately motivated its selection for the prototypes. Further details on iMarker fabrication, their distinctions from existing state-of-the-art blending markers, and the associated pragmatic design considerations are thoroughly presented in **Paper#1** (§4).

2.1.2 Contributions

Detecting iMarkers requires a fundamentally different approach than that for classic printed fiducial markers. Because iMarkers are visible only under specific optical conditions, the detection pipeline must address two complementary challenges:

1. Designing appropriate **visual sensors** capable of revealing CSR-based reflections of iMarkers, and



(a) 4×4 (b) 5×5 (c) 7×7 (d) MIP_36h12 (e) Original

Figure 2.2: Some instances of ArUco marker [82] dictionary variants with marker_id = 5.

2. Developing **software algorithms** that reliably extract marker patterns and decode their pose from these sensor outputs.

Accordingly, a contribution of the thesis is to develop a complete **end-to-end iMarker detection stack**, spanning custom hardware, tailored algorithms, and integration into robotic systems, addressing the concerns of **RQ1** (§1.4.2).

On the hardware side, the sensing solution must remain *affordable*, rely on *standard optical components*, and be *simple* enough to construct and iterate on, especially given the project's early experimental phase. At its core, the detector relies on low-cost, standard USB cameras with a suitable Field of View (FoV), plug-and-play operation, and strong software support in robotics. Therefore, these baseline cameras are augmented with additional optical elements introduced in §1.2.1, including polarizers, beam splitters, and illumination modules, resulting in a set of custom sensor prototypes capable of capturing and maximizing the distinctive reflection bands of CSRs in iMarkers.

On the software side, the iMarker detection pipeline must deliver robust, real-time performance, enabling the system to **localize potential iMarkers** and **extract their pose and patterns** within milliseconds, an essential requirement for robotics and AR applications. Therefore, the algorithm pipeline includes iMarker sensor data acquisition, frame analysis, iMarker detection and decoding, pattern extraction, and pose estimation.

In practice, and through close interdisciplinary collaboration within the team, two variants of iMarker detection sensors are proposed, each aligned with a different software processing pipeline. These designs fall into two categories [164]:

- **Dual-vision setups**, which employ two synchronized cameras with differently oriented polarizers observing opposite sides of a beamsplitter; and
- **Single-vision setups**, which use a single camera equipped with either a fixed or switchable polarizer.

As discussed in §1.2.1, the role of the polarizers is to block the CSR-filled regions of iMarkers. Figure 2.3 illustrates these hardware configurations.

The algorithmic pipelines (regardless of the sensor variant used) begin by acquiring raw sensor inputs (*i.e.,* frames) and applying a series of preprocessing steps to enhance frame quality. The subsequent detection stage differs depending on the sensor configuration:

- In the **dual-vision setup**, the pipeline ensures synchronized image capture, performs spatial alignment via feature matching, and subtracts the two polarization-separated views to reveal CSR regions.
- For the **single-vision setup**, two variants exist:



Figure 2.3: Variant iMarker detection sensors (taken from [164]), including the dual-vision setup with cube **(a)** and plate **(b)** beamsplitters, and single-vision setups with static **(c)** and dynamic **(d)** polarizers.

- With a **fixed polarizer**, CSR regions appear consistently blocked, enabling segmentation and detection through simple color masking.
- With a **switchable polarizer**, marker regions are recovered using *temporal frame subtraction* between alternating polarization states.

In all cases, the extracted marker pattern masks are ultimately processed by the ArUco detector [82], together with camera intrinsics, to estimate marker poses.

To evaluate the proposed end-to-end iMarker detection stack, the author developed a **general software framework** [165]. The software is fully open-source and provides both a **standalone variant**[2], featuring an intuitive Graphical User Interface (GUI) for configuring parameters and visualizing detections, and a **Robot Operating System (ROS)-based variant**[3] for straightforward integration into robotic pipelines.

Further details on iMarker detection, spanning the detailed sensor designs to implemented algorithm evaluations, are thoroughly presented in **Paper#1** (§4). For additional material, including hardware schematics, demonstration videos, and supplementary resources, a dedicated project webpage is available at `https://snt-arg.github.io/iMarkers/`.

[2] `https://github.com/snt-arg/iMarker_detector_standalone`
[3] `https://github.com/snt-arg/iMarker_detector_ros`

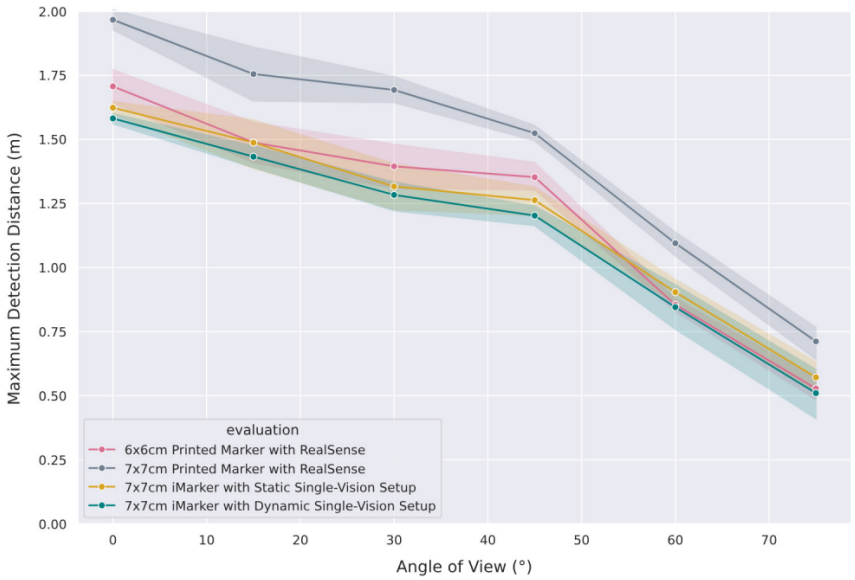


Figure 2.4: Maximum detection distances for "printed" and "iMarkers" (ArUco patterns) at various viewing angles (taken from [164]).

2.1.3 Summary of Results

Evaluations of the effectiveness of iMarkers against their printed counterparts, with a focus on the **maximum detection range** across varying viewing angles and **pose estimation accuracy**, prove the applicability of iMarkers. As shown in Figure 2.4, the single-vision setup with static/dynamic polarizers was used to read printed and iMarkers with equal dimensions, and a motion capture system was used to provide ground truth. It can be seen that the maximum detection range decreases with increasing viewing angle in all cases due to perspective distortion. Static and dynamic single-vision iMarker setups exhibit nearly identical detection ranges, differing by only 3 cm to 7 cm on average, while printed ArUco markers consistently achieve longer ranges owing to their high-contrast patterns. Nevertheless, iMarkers demonstrate comparable real-world performance while remaining visually invisible, making them well-suited for short-range, aesthetics-sensitive applications. Additionally, another experiment revealed that pose estimation accuracy remains consistent between iMarkers and printed ArUco markers, with mean errors increasing slightly at larger viewing angles due to perspective distortion and relative differences between marker types staying within 1.3% to 3.9%. This consistency arises from the hardware–software design, which produces pre-processed binary images for reliable detection in both marker types. While printed markers inherently offer higher detectability, iMarkers provide a visually imperceptible alternative that maintains competitive pose accuracy and detection performance for short-range applications.

Another evaluation focused on recognition speed, which was conducted by a profiling analysis. The results in Table 2.1 show that the complete iMarker detection and recognition pipeline (from frame acquisition to pattern decoding) runs in real time and adds only about $\approx 100\ ms$ overhead to the standard ArUco library using CPU-only processing. Among the evaluated configurations, UV-range iMarkers achieve the fastest recognition due to minimal preprocessing, while dual-vision setups incur higher acquisition and processing costs from the need for synchronized capture and image alignment. Overall, all configurations meet real-time requirements and remain significantly more efficient than several existing fiducial marker systems that take seconds to detect.

Table 2.1: Performance analysis of iMarkers detection and recognition, based on cumulative time profiling of required modules (averaged over 40 frames) in milliseconds (ms) (taken from [164]). SV, DV, UV, and BS indicate single-vision, dual-vision, ultraviolet, and beamsplitter, respectively.

	iMarker Variant				
	UV	*Visible-range*			
Software Module	**SV Static**	**SV Static**	**SV Dynamic**	**DV (cube BS)**	**DV (plate BS)**
Acquisition	11 ms	15 ms	13 ms	32 ms	36 ms
Preprocessing	2 ms	7 ms	9 ms	14 ms	19 ms
Processing	3 ms	23 ms	29 ms	39 ms	42 ms
Post-processing	3 ms	9 ms	8 ms	4 ms	5 ms
Recognition	2 ms	2 ms	3 ms	2 ms	3 ms
Sum	21 ms	56 ms	62 ms	91 ms	105 ms

2.2 State-of-the-art Visual SLAM Approaches (RQ2)

2.2.1 Overview

To investigate the applicability of iMarkers in a realistic and impactful setting, a suitable **case study** was needed, and **Visual SLAM (VSLAM)** was selected due to its reliance on **visual features**. This choice allows us to evaluate how introducing additional landmarks (such as fiducial marker variants) can enhance robots' understanding of the environments. To build a solid foundation and address the concerns of **RQ2** (§1.4.2), a comprehensive survey of VSLAM methods, along with trends and state-of-the-art solutions, was conducted. According to the survey **Paper#2** (§5), VSLAM has evolved considerably over time, spanning vision sensor setups, benchmark datasets, and operating conditions. Its wide methodological variations and importance for robust localization and mapping have made it one of the most active research areas in robotics. The survey studies recent influential VSLAM works, analyzing them across *novelty domains*, *objectives*, *algorithmic strategies*, and *semantic integration levels* [166]. It further identifies current *trends*, *open challenges*, and *emerging directions* in the field. This study provides a consolidated understanding of modern VSLAM developments and serves as a foundation for the subsequent design of our marker-enhanced VSLAM framework.

2.2.2 Contributions

The survey begins with a **general overview** of a standard VSLAM system, focusing on the *core methodologies* and *functional roles* of its primary modules. It also reviews existing literature and **prior surveys** in the VSLAM domain, positioning the present work within the robotics community and highlighting gaps that earlier surveys do not fully address. Furthermore, the survey traces the **evolution of VSLAM algorithms**, from the first monocular systems that laid the foundations of the field to later reliable baselines. In addition, it investigates the wide range of **configurations** adopted in modern VSLAM systems, including sensor and data acquisition setups, visual feature processing methods, target environments, evaluation protocols, and the integration of semantic-level information across modules.

The survey reviews fifty impactful state-of-the-art works and categorizes them based on their primary contributions to the domain as follows:

- *Multi-sensor processing*, covering approaches that employ multiple sensors to im-

prove environmental understanding, including multi-camera and multi-modal configurations.

- *Pose estimation accuracy enhancement*, containing methods aimed at improving localization accuracy through the use of points and lines, fiducial markers, or deep learning–based pose estimation techniques.
- *Real-world viability*, stating systems designed to operate robustly across diverse environments and challenging real-world conditions.
- *Resource constraint solutions*, including methodologies tailored for platforms with limited computational resources, such as lightweight pipelines or computation offloading strategies.
- *Versatility*, covering frameworks emphasizing ease of development, adaptability, and extensibility to facilitate rapid integration of new functionalities.
- *Visual odometry*, containing works primarily focused on estimating the robot's position and orientation with the highest possible accuracy.

Considering the identified categories, the survey analyzes the **current trends** in the VSLAM community by focusing on the shared aspects of the reviewed papers. In addition, it discusses emerging **research directions** in contemporary VSLAM methods and summarizes widely adopted evaluation practices, including benchmark datasets and testing environments.

2.2.3 Summary of Results

As shown in Fig. 2.5, the reviewed works were analyzed to extract current trends in VSLAM research, including the choice of baseline frameworks **(a)**, system objectives **(b)**, evaluation datasets **(c)**, incorporation of semantic information **(d)**, support for dynamic or static scenarios **(e)**, and target environments **(f)**. The results show that most systems are developed as standalone frameworks, with ORB-SLAM variants being the most commonly adopted baselines for new VSLAM developments. With respect to system objectives, recent works predominantly focus on improving the visual odometry (front-end) stage, followed by enhancing real-world robustness to handle dynamic and challenging environments. However, resource efficiency and versatility receive comparatively less attention. In terms of evaluation, TUM RGB-D [167] remains the most widely used benchmark due to its diverse set of scenarios, while many studies rely on in-house datasets tailored to specific application needs, reflecting a growing trend toward real-world validation. Regarding semantic integration, most surveyed approaches do not explicitly incorporate semantic information, likely due to the additional computational cost and the risk of propagating incorrect semantic cues. This trend aligns with the observation that more than half of the systems are designed to operate in dynamic environments rather than assuming static conditions. Finally, most VSLAM methods are evaluated in indoor settings or across both indoor and outdoor environments, with comparatively fewer works focusing exclusively on outdoor scenarios.

Based on the analysis of state-of-the-art VSLAM systems, several promising research directions emerge that are likely to shape the future evolution of visual SLAM. Although significant progress has been achieved across core modules such as tracking, mapping, and loop closing, open challenges remain in balancing *robustness*, *accuracy*, and *real-time*

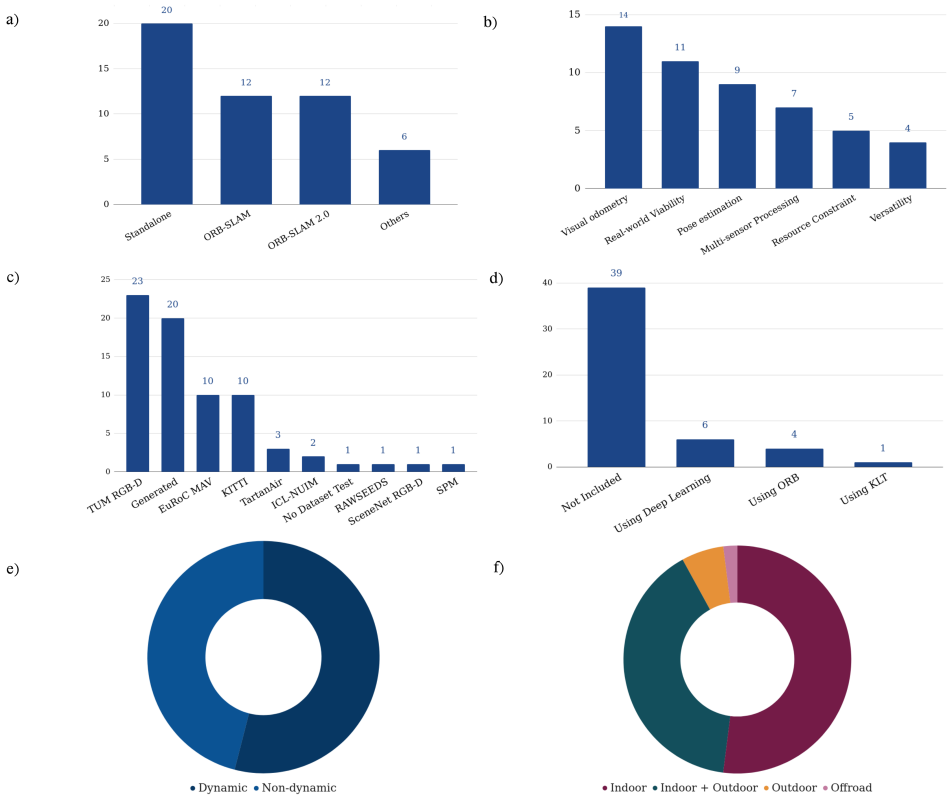


Figure 2.5: The current trends of VSLAM approaches (taken from [166]): (**a**) base SLAM system employed to implement a new approach, (**b**) the primary objective of the approach, (**c**) various datasets that the proposed methods were tested on, (**d**) the impact of utilizing semantic data in the proposed methods, (**e**) the amount of dynamic objects existing in the environment, (**f**) various types of environments the system tested on.

performance. Accordingly, one prominent direction is the increasing adoption of learning-based feature processing, where DNNs are used for feature extraction, object detection, and semantic understanding. This can be extended to enhance visual odometry and loop closure performance, despite ongoing concerns regarding generalization, misclassification, and dynamic object handling. Another related topic is the *trade-off* between information richness and computational cost While dense scene representations provide comprehensive environmental detail, they remain difficult to process in real time. On the other hand, sparse methods often sacrifice contextual completeness and achieve efficient modular balance through the VSLAM pipelines.

Another major trend is the integration of *semantic information* into mapping and localization, enabling higher-level scene understanding that benefits pose estimation, planning, and data association. Additionally, *loop closure detection* remains a critical bottleneck as map size increases, motivating further research into efficient image retrieval, feature aggregation, and optimization strategies. Finally, robustness in *challenging scenarios* remains another key objective, encouraging the incorporation of complementary cues, including structural primitives, object-level representations, and alternative landmarks. As discussed in the survey, these directions highlight a shift toward more semantically aware, computationally efficient, and environment-robust VSLAM systems that can operate reliably in real-world conditions.

2.3 Marker-Enhanced Perception (RQ3)

2.3.1 Overview

Due to the interdisciplinary nature of the project, the idea was to develop a **printed fiducial marker-based robotics framework** in parallel with the ongoing iMarker fabrication investigations and efforts. This allowed the robotic integration pipeline to be designed, tested, and validated early on, while ensuring that the entire framework could later be adapted to iMarkers with minimal modifications once the sensing and detection capabilities matured. Given the characteristics of VSLAM systems, which employ low-cost hardware to provide rich visual information, they are good candidates for integrating fiducial markers in robotics. For this purpose, fiducial markers (whether traditional printed ones or iMarker versions) can encode rich environmental information, aiding map reconstruction. In this regard, available marker-based VSLAM approaches primarily use markers to improve *visual feature extraction* in low-feature environments and to *generate low-level geometric maps*. Another use case is revisiting the same marker and triggering a loop-closure detection flag, thereby improving localization, pose estimation, and map reconstruction accuracy. Thus, current marker-based VSLAM frameworks use fiducial markers as additional landmarks for tracking, but they do not **encode meaningful semantic information** within the markers and employ them solely in purely geometric map representations. This leaves a clear gap in fully using the potential of fiducial markers to accurately identify the *semantic elements* and the *topological relationships* among them.

2.3.2 Contributions

To fill in the mentioned gaps and address the concerns of **RQ3** (§1.4.2), a marker-based VSLAM system was developed and presented in **Paper#3** (§6). This system builds upon **UcoSLAM** [150], a purely geometric VSLAM framework that employs ArUco markers [82] for map consistency, data association, and loop-closure detection. The proposed extension, named **Semantic UcoSLAM** [51], incorporates semantic augmentation by using fiducial markers to label the *walls* of indoor environments. Accordingly, each wall in the environment is tagged with a unique ArUco marker, and a pre-defined database (stored as a JSON file) maintains these associations. As the markers are strategically placed on walls, their equations are inferred from the associated marker poses and updated only when a marker appears in a recent KeyFrame. For example, the database specifies that marker_id = 5 and marker_id = 6 belong to the same corridor. This enables the system to later infer higher-level structures, such as *two-wall rooms (corridors)* and *four-wall rooms*, from marker data. This is fed as prior information to the autonomous robot equipped with a visual sensor and running the *Semantic UcoSLAM* system while navigating in an environment. Whenever the visual odometry module observes a fiducial marker within the camera's FoV, it recognizes its unique pattern and identifier, estimates its pose, and adds the marker to the map, constraining the detected frame. Then, when all fiducial markers belonging to the same corridor or room are recognized and mapped, a corresponding semantic entity is instantiated within the factor graph. The **poses** of individual walls and the **centroids** of higher-level entities are continuously refined through back-end optimization and bundle adjustment. Loop detection and correction are also marker-driven. When a previously seen marker is revisited, a constraint is added between the current and past poses, allowing the optimizer to redistribute accumulated drift and correct both the map and trajectory.

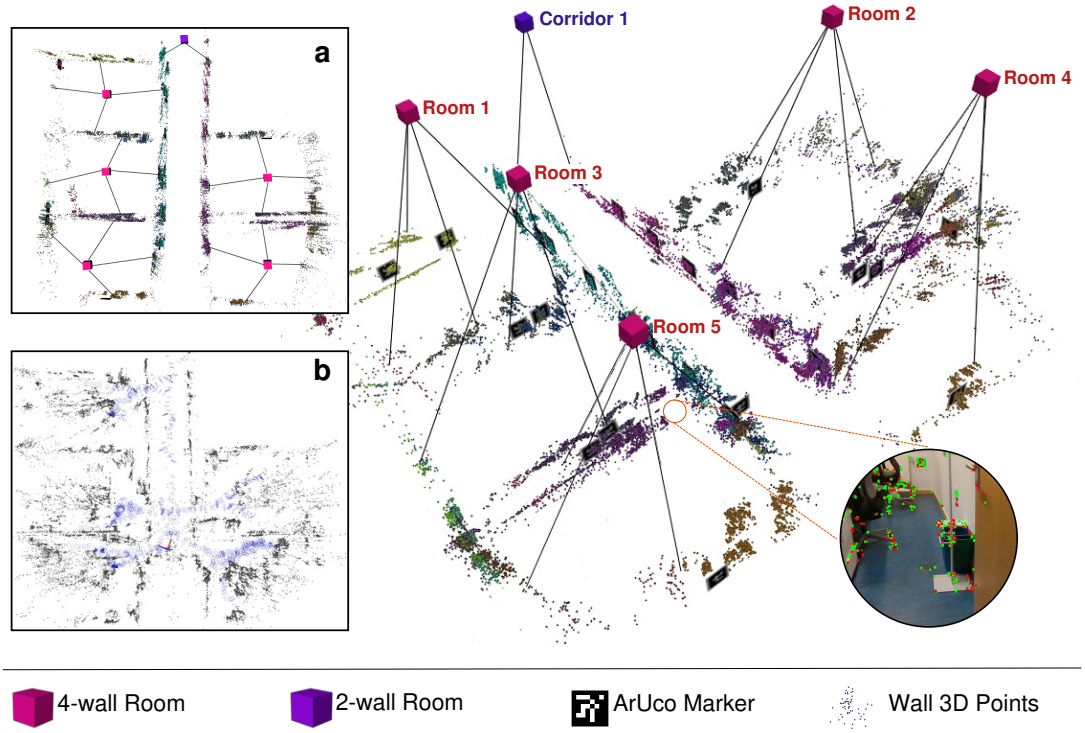
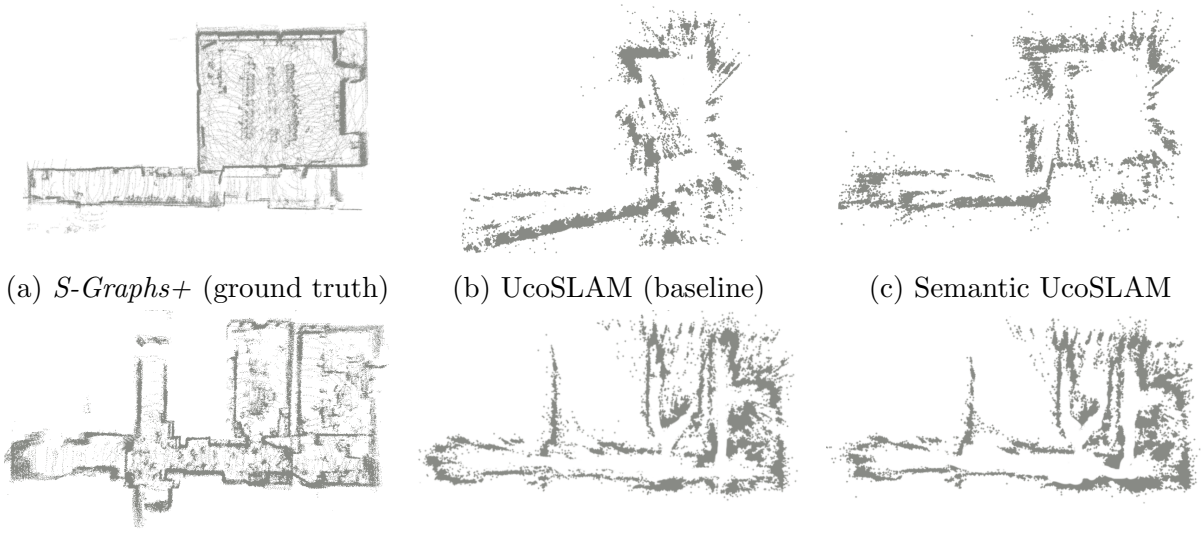


Figure 2.6: A hierarchical representation of the reconstruction map using the proposed method (taken from [51]): **a)** the top view of the reconstructed map represented in 2D, **b)** keypoints and robot trajectory records that lead to map reconstruction.

Thus, the primary contribution of the Semantic UcoSLAM framework is the addition of a **Semantic Analyzer** module, which plays a crucial role in integrating the detected markers (and their attached walls) into the map. This module continuously performs *wall* and *room detection* pipelines using the mapped points and marker poses from the tracking stage. The mapping module handles newly added vertices and edges, improving the final map for optimized reconstruction. Finally, a marker-based loop detection checks whether the markers in the current frame have been seen before and, if so, triggers the global optimizer pipeline to enhance the reconstructed map.

To support the goals of the proposed framework, a new set of geometric constraints has been added to the baseline, including *marker-to-frame*, *marker-to-wall*, and *wall-to-room*. For *marker-to-frame* constraints, the system establishes an edge linking the *frame vertex* in which an ArUco marker is observed to the corresponding *marker vertex*, encoding both the marker identifier and its estimated pose. For the detected walls, their vertex nodes are factored into the system using their azimuth and elevation in the global reference frame. Accordingly, the *marker-to-wall* edge connects each detected and mapped fiducial marker to its associated wall, enforcing a constraint that minimizes the discrepancy between the wall's azimuth angle and the marker's inferred orientation. To enable room-level reasoning, the framework proposes a *wall-to-room* constraint, driven by a lightweight semantic loader that associates rooms with their corresponding fiducial marker identifiers in the database, without requiring explicit pose information. Building on a modified *S-Graphs* formulation, detected walls are linked to common marker sets to define room entities, with two facing parallel walls reflecting a *corridor* and two pairs of parallel walls forming a *four-wall room*. The wall-to-room constraints enforce structural consistency across associated walls and stabilize the estimated room layout, enabling robust room



(a) *S-Graphs+* (ground truth) (b) UcoSLAM (baseline) (c) Semantic UcoSLAM

(d) *S-Graphs+* (ground truth) (e) UcoSLAM (baseline) (f) Semantic UcoSLAM

Figure 2.7: Reconstructed maps using Semantic UcoSLAM, the baseline (UcoSLAM), and the LiDAR ground truth on two different instances of the collected dataset (taken from [51]).

segmentation even under partial observability.

Moreover, another contribution of the work is to represent the environment as an optimizable 3D scene graph, inspired by LiDAR-based S-Graphs [155]. For this purpose, the locations of walls and room variants are calculated from markers rather than odometry readings and from planar surface extraction from LiDAR scans. In this regard, Figure 2.6 depicts the hierarchical representation of the reconstruction map using the proposed Semantic UcoSLAM system.

2.3.3 Summary of Results

For evaluation, an in-house real-world dataset was collected using a legged robot equipped with a LiDAR sensor as ground truth and a monocular camera for visual perception. The robot operated in different office zones with various corridors and room layouts, with the walls labeled with printed ArUco markers. The Absolute Trajectory Error (ATE) measurements (mainly Root Mean Square Deviation (RMSE) and Standard Deviation (STD)) were used to compare the performance of the Semantic UcoSLAM to its baseline and LiDAR ground truth.

Experimental results showed that the proposed system outperforms its baseline in accuracy, even with the addition of new constraints, yielding enhanced map representations. Figure 2.7 depicts the positive impact of employing *marker-to-wall* and *wall-to-room* constraints on mapping performance. This improvement stems from the proposed method's ability to introduce additional geometric and semantic constraints into the map, owing to the new constraints and associations among semantic entities. The proposed system clearly outperforms the baseline in challenging scenarios where conventional keypoint- or marker-based map information fetching fails. Further details on the framework design and the impact of markers for better environment understanding are presented in **Paper#3** (§6).

2.4 iMarker-integrated VSLAM (RQ4)

2.4.1 Overview

Once a **marker-based VSLAM** framework had been established, and in parallel, the **iMarker detectors** and initial **iMarker prototypes** were being developed, the next step was to evaluate whether iMarkers could be integrated into the VSLAM pipeline and fulfill the same functional role as conventional fiducial markers. To pursue this objective, the baseline framework was transitioned to **ORB-SLAM 3.0** [108], a more modern and versatile VSLAM system. It supports a broader range of *visual* and *visual–inertial* sensor configurations, offers improved accuracy, and aligns more closely with current state-of-the-art practices in VSLAM. This update enabled a more realistic evaluation of whether iMarkers can be integrated into a classic VSLAM pipeline and perform comparably to printed fiducial markers.

Fiducial markers serve as essential reference anchors that enable the system to contextualize semantic information within the environment. Notably, the markers themselves encode no intrinsic semantic or geometric attributes, such as global pose or physical dimensions. The new framework uses fiducial markers placed on *walls* and *doorways* and infers higher-level environment-driven entities, including *corridors* and *rooms*. Accordingly, each room or corridor consists of a doorway and multiple walls, where a database of marker_id association is fed to the system, similar to **Semantic UcoSLAM** [51]. The marker poses used to label walls and doorways correspond to their positions at mapping time. The strategy for detecting environment-driven entities is inspired by *S-Graphs* [155], where a *corridor* is characterized as a space bounded by two parallel walls (with facing normal vectors), while a *room* (typically rectangular) comprises four walls, forming two pairs of parallel walls perpendicular to each other. Consequently, even without relying on deep learning-based semantic segmentation, the system recovers meaningful semantic structure solely from marker geometry, enabling fast, lightweight map enrichment. Owing to the baseline, this framework also contains a map manager thread for creating, merging, and handling reconstructed maps. Additionally, unlike Semantic UcoSLAM, which supports monocular input, this framework leverages RGB-D data for improved depth extraction and robust mapping.

2.4.2 Contributions

The primary contribution of this work is the generation of human-comprehensible **digital twins** of indoor environments using multi-layered, hierarchical **scene graphs**. The paper proposes a framework that provides comprehensive robotic situational awareness by combining geometric maps with hierarchically organized semantic entities and their topological relationships into a single graph, improving SLAM performance through semantic exploitation. In the proposed framework, the custom scene graph organizes information across three layers: KeyFrames, fiducial markers, and map points at the lowest layer; walls and doorways at the intermediate layer; and high-level environment entities such as rooms and corridors at the highest layer. Figure 2.8 depicts an example of a generated three-layered graph for an indoor environment, interconnecting the robot poses with diverse structural and geometric priors about the scene. The proposed framework tightly couples the robot poses with structural semantic entities identified by fiducial markers within a multi-layered, hierarchical, and optimizable graph.

Another contribution contains a set of semantic-geometric constraints. The first one is the *marker-to-frame* constraint, which is mainly used to detect structural-level and seman-

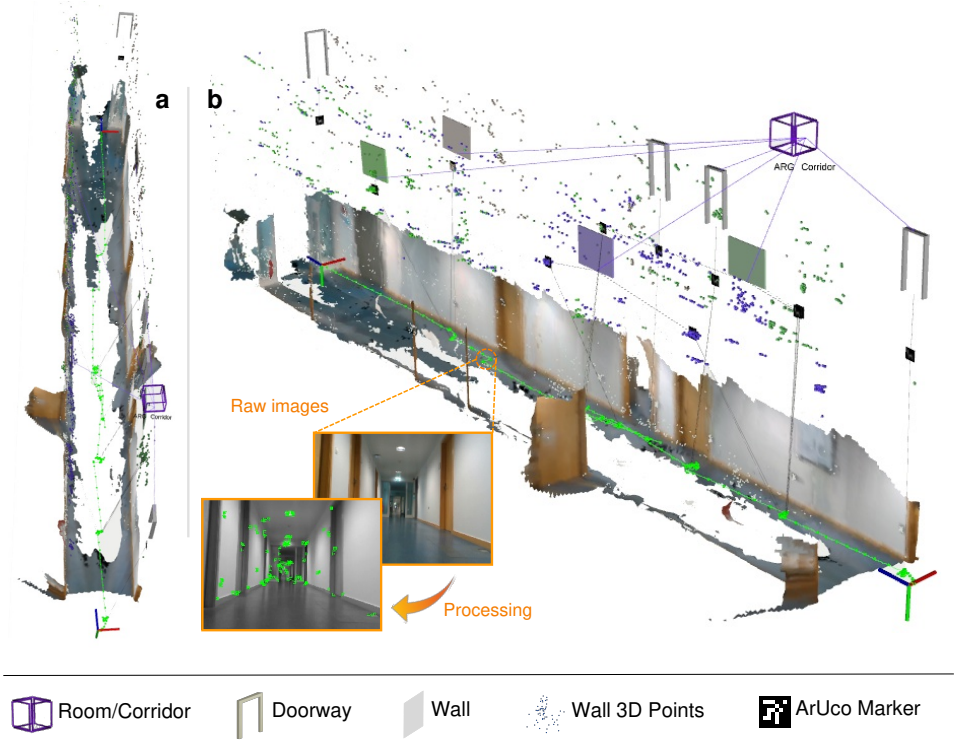


Figure 2.8: A reconstructed map of an indoor environment with its hierarchical representation generated by the proposed framework, containing various detected structural-level entities and the connections among them (taken from [52]): (**a**) the top view of the reconstructed map. (**b**) The final generated 3D view.

tic entities (in this work, walls and doorways). In this paper, as in Semantic UcoSLAM, markers are assumed to be placed directly on walls or the inner parts of door frames. Each marker is constrained by the observing KeyFrame through its locally estimated pose, with the association modeled using a Mahalanobis distance weighted by the corresponding information matrix. Similar to Semantic UcoSLAM, each wall is modeled as a planar entity in the global reference frame and linked to its attached fiducial marker through a geometric *marker-to-wall* constraint that enforces consistency in orientation and position. It ensures that the wall's estimated orientation aligns with the marker's pose and that the marker lies on the wall surface. The detection of doorways is performed in a similar way and by incorporating *marker-to-doorway* constraint. Other semantic constraints are *wall-to-corridor* and *wall-to-room*, inspired by the Semantic UcoSLAM approach. Corridors contain two parallel wall planes aligned with the x-axis and a centroid point, in which the normal vectors of the walls are facing each other. Thus, the cost function of the constraint minimizes the corridor's vertex node and its corresponding wall planes. Finally, the *wall-to-room* constraint applies to four walls facing each other and a centroid point.

Moreover, this paper represents the first integration of *iMarkers* into the robotics domain. The paper demonstrates that iMarkers can replace traditional fiducial markers to identify and localize structural-level semantic entities, such as walls and doorways. This can simplify the establishment of meaningful associations with higher-level entities, including corridors and rooms. It should be noted that, due to the limited number of fabricated samples, iMarkers were used selectively (primarily for labeling door frames) as a **case study** to demonstrate that they can provide equivalent geometric information

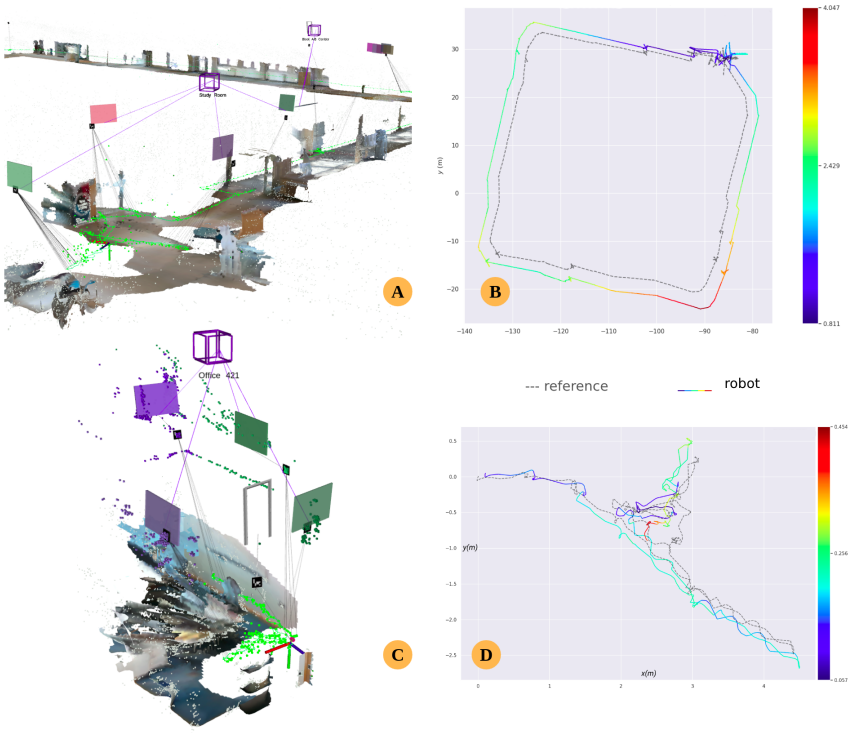


Figure 2.9: The qualitative results and Absolute Trajectory Error (ATE) of the proposed approach *w.r.t.* translation in *meters* on the real-world data instances. The dotted lines in the charts are the LiDAR ground-truth reference values, and the solid lines show the robot's trajectory.

to conventional fiducial markers. Accordingly, this work directly addresses the concerns of §**RQ4** (1.4.2), namely the potential of iMarkers in robotics applications. Further details on the updated framework design and the integration of markers and iMarkers are presented in **Paper#4** (§7). The proposed framework is publicly available in `https://github.com/snt-arg/visual_sgraphs/tree/v1.0.0`.

2.4.3 Summary of Results

According to the experimental results on a real-world dataset collected using legged robots, incorporating the introduced constraints improves map organization and consistency within the factor graph. Furthermore, the proposed framework excels at crafting a richer, multi-layered hierarchical map while simultaneously enhancing robot pose accuracy. Figure 2.8 depicts the 3D scene graphs of the environment with detected semantic entities, plus the generated trajectories. Performance analysis showed that the proposed method consistently outperforms the baseline and competing frameworks across most sequences, primarily because it introduces structural and semantic constraints into the map. While the baseline (ORB-SLAM 3.0) slightly outperforms the proposed method on some sequences by a negligible margin, likely due to marker noise under varying lighting conditions, the proposed method additionally delivers a three-layered situational graph.

In addition, all evaluation data were recorded at 25fps, and validation results confirm that the proposed multi-threaded architecture maintains real-time performance at this rate. The computational load is unevenly distributed across modules: while tracking and marker detection run continuously and efficiently, more expensive processes such as optimization, loop and map merging, and global bundle adjustment are executed asyn-

chronously and only when required, preventing any impact on the frame rate. Structural and semantic entity extraction incurs minimal overhead, accounting for only 10-15% of the total processing time. Although the proposed framework introduces additional constraints and thus operates at a higher average cost than ORB-SLAM 3.0, it maintains low latency and real-time performance, highlighting a deliberate trade-off between computational cost and richer, semantically informed map representations.

Finally, the paper demonstrated, for the first time, the applicability of iMarkers in robotics, labeling door frames with prototype transparent iMarkers. Since iMarkers differ from printed ArUco markers only at the detection stage, the generated semantic map remains unchanged, confirming their seamless integration into the framework.

2.5 Minimal-Landmark Situational Awareness (RQ5)

2.5.1 Overview

The final (and most significant) contribution of this thesis is the design of a robust VSLAM framework that overcomes the limitations of previous methodologies, which were highly dependent on the presence of fiducial markers (and iMarkers). This framework, named **vS-Graphs** [168], tightly couples visual perception, SLAM, and semantic scene understanding by extending and improving the structure introduced earlier in §2.4. It employs RGB-D data for the classical VSLAM pipeline, including tracking, mapping, local and global bundle adjustment, and loop closure detection. In addition, two new parallel threads are introduced: one that detects **building components** (including walls and ground surfaces), and another that infers **structural entities** (rooms and floors) from these components and integrates them into a hierarchical representation. All outputs are encoded in an optimizable 3D scene graph that is tightly coupled with the VSLAM pipeline. The scene graph is organized into three levels: the lowest layer contains KeyFrames, optional fiducial markers, 3D map points, and point clouds; the middle layer represents building components; and the highest layer models structural elements of the environment.

vS-Graphs is designed as a *marker-optional* system, allowing the robot to understand its surroundings without relying on artificial landmarks. Thus, fiducial markers and iMarkers act as supplementary cues rather than dependencies. While the framework does not rely on markers for mapping accuracy or robustness, it uses them to infuse additional semantic information that enhances environmental understanding. For example, marker_id = 5 may be associated with *corridor-A*, while marker_id = 8 may correspond to *office-B*, enriching the 3D scene graph with human-interpretable labels. They also laid the foundation for future works, including high-level semantic loop closure detection (*i.e.,* revisiting the same room based on its positioned marker). Figure 2.10 illustrates an indoor environment model generated by vS-Graphs.

Accordingly, this work directly addresses the concerns of **RQ5** (1.4.2), where artificial landmarks provide supplementary aids to enhance robotics perception. Further details on the vS-Graphs framework design and the role of markers and iMarkers are presented in **Paper#5** (§8). The proposed framework is publicly available in `https://github.com/snt-arg/visual_sgraphs`.

2.5.2 Contributions

The proposed framework is inspired by LiDAR-based *S-Graphs* [156] and seamlessly integrates scene graph generation into the VSLAM procedure, exploiting the richer semantic

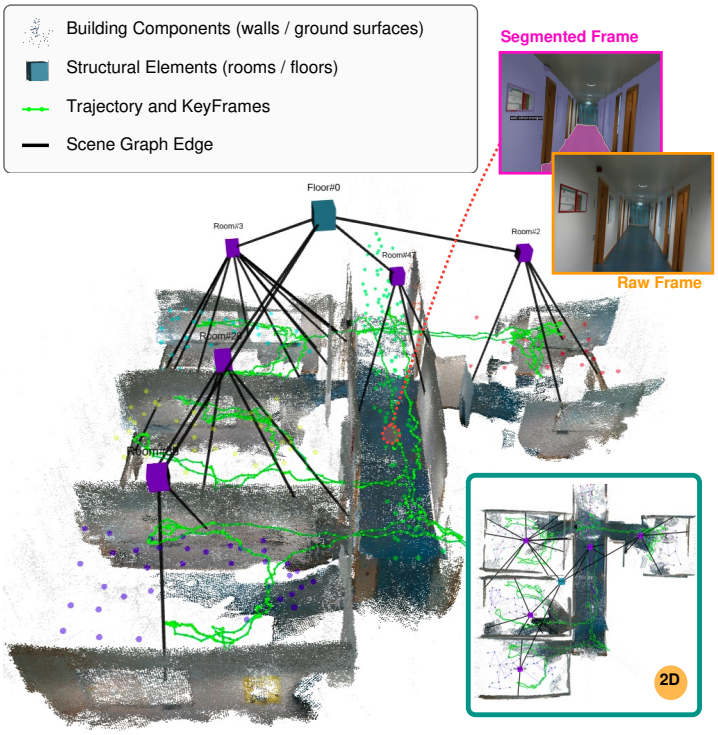


Figure 2.10: A reconstructed map tailored to the optimizable 3D scene graph generated by the proposed vS-Graphs, enriched with environment-driven semantic entities (taken from [168]).

and appearance information in visual data to both validate and contextualize structural elements. In this regard, the system's contributions can be summarized as: **(a)** the design of a modular and extensible system architecture, **(b)** a robust method for recognizing and mapping fundamental building components, **(c)** a mechanism for extracting high-level structural elements from them, and **(d)** generating interpretable 3D scene graphs with hierarchical optimization capabilities.

Regarding the system design, while vS-Graphs is built on ORB-SLAM 3.0, it modifies its core modules and adds new threads for robust scene analysis and map reconstruction. The primary modifications occurred during the integration of two new threads: one responsible for recognizing building components and the other for extracting structural elements. Other modules of the baseline were modified accordingly to support localization, mapping, and the optimization of these elements. Furthermore, the framework supports the recognition of fiducial markers (and potentially iMarkers) to augment the information describing these environment-driven semantic entities.

For building component recognition, particularly the accurate detection of fundamental environment-driven elements (*i.e.,* ground and wall surfaces) within the camera's field of view, the framework incorporates a deep learning–based scene segmentation module. For this, the module processes the point cloud and visual data obtained from KeyFrames to improve scene understanding. It processes incoming RGB data of the KeyFrames to isolate wall and ground regions and discard irrelevant entities, generates separate point clouds for these components, and filters the raw depth data accordingly to retain only structurally relevant elements. Subsequently, plane-fitting and geometric verification algorithms are applied to validate the extracted planar entities and estimate their 3D poses, enabling the reliable integration of building components into the mapping and optimiza-

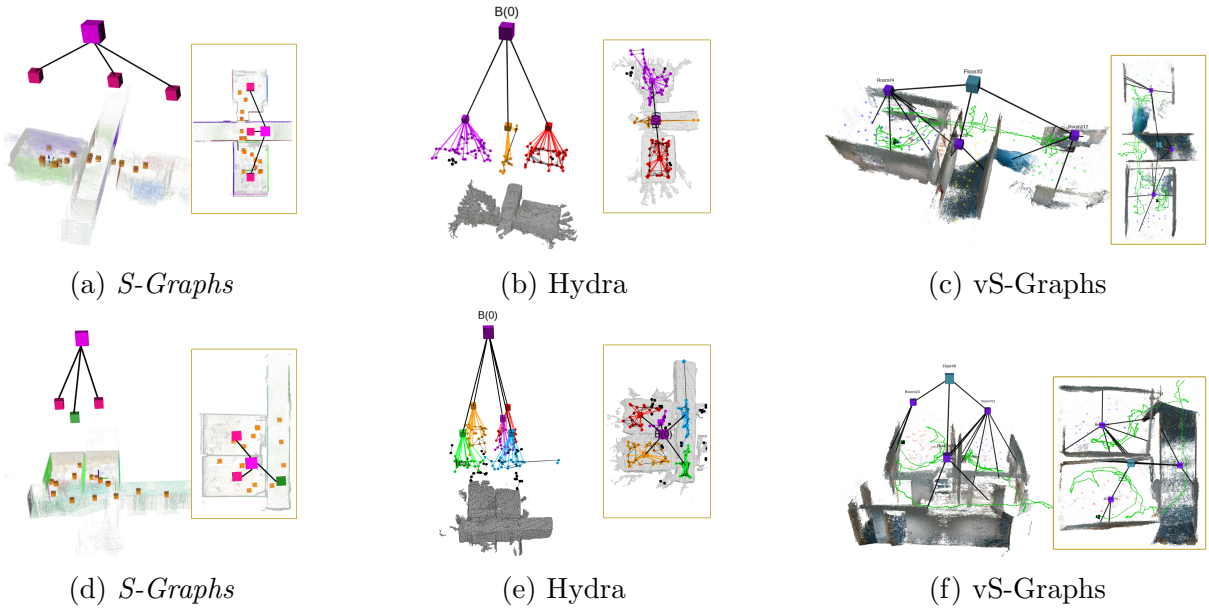
tion pipeline. The last step is to check the detected building components against structural validation rules, ensuring they satisfy geometric constraints for the environment. Once the building components in each KeyFrame are recognized, the mapping and optimization pipelines incorporate them as constraints, ensuring their accurate localization and consistent integration within the reconstructed map. Constraints at this level enforce the wall's orientation and position by preserving its normal vector and plane offset, while a configurable threshold limits the allowable distance between the wall and surrounding free-space points. The optimization updates each room's centroid based on its corresponding walls, identifying parallel and perpendicular wall pairs within the room.

For structural element recognition, a dedicated thread runs at fixed intervals, extracting rooms and floor entities from the building components of the current active map. The thread actively analyzes mapped building components while using free-space information from the depth data to identify potential room layouts, where a room is a spatially enclosed area bounded by at least two walls that enclose a contiguous free-space region on the ground plane. The optimization updates each room's centroid based on its walls, identifying parallel and perpendicular wall pairs. Additionally, a floor is treated as a higher-level structural element, representing a collection of rooms within the same building level. It should be noted that the current version of vS-Graphs supports only single-floor mapping.

2.5.3 Summary of Results

To validate the idea of vS-Graphs as a robotics framework for improving situational awareness, several experiments have been conducted. Accordingly, the system has been tested on a wide range of standard benchmarks and in-house datasets, and compared against widely adopted VSLAM systems, including the baseline.

Assessing trajectory estimation accuracy showed that vS-Graphs consistently outperformed other state-of-the-art works, thanks to the additional constraints imposed by localized elements. Furthermore, incorporating loop closures and room–wall constraints further improves the performance in multi-room and looped sequences. Considering map-



(a) *S-Graphs* (b) Hydra (c) vS-Graphs

(d) *S-Graphs* (e) Hydra (f) vS-Graphs

Figure 2.11: Qualitative comparison of reconstructed scene graphs on multi-room sequences of the collected dataset.

ping performance, the accuracy of the reconstructed vS-Graph maps against LiDAR-based ground truth is more robust than that of the baseline. vS-Graphs achieves higher mapping accuracy despite producing maps with $\approx 10.5\%$ fewer points on average than the baseline, thanks to environment-aware constraints that ensure a more coherent and consistent reconstruction.

Another conducted experiment was runtime analysis, which proved that vS-Graphs achieves real-time performance, surpassing the 20 FPS threshold for real-time operation, thanks to its multi-threaded architecture. Profiling results showed that the local mapping and optimization pipelines consume more resources than the continuously running tracking thread, while the newly incorporated threads for building component and structural element recognition do not create a bottleneck for the system.

Finally, a scene understanding evaluation was conducted for vS-Graphs and other state-of-the-art systems, focusing on detecting entities essential for structural layout interpretation. Accordingly, multi-room sequences from the collected dataset with LiDAR-based ground truth were used to benchmark. In this regard, despite relying solely on RGB-D input, vS-Graphs achieves accuracy comparable to LiDAR-based S-Graphs and demonstrates higher recall in complex environments thanks to visual verification of building components. Qualitative comparisons in Figure 2.11 further highlight vS-Graphs' ability to reconstruct consistent and precise structural scene graphs.

3

Conclusions

3.1 Limitations and Open Challenges

Despite the achievements and promising results presented in this thesis, several limitations must be acknowledged. These limitations arise primarily from time constraints, resource availability, and the interdisciplinary nature of the proposed contributions, which restricted a more exhaustive investigation of certain components.

The identified limitations are discussed from two complementary yet fundamentally different perspectives, reflecting the distinct nature of the proposed contributions. The first perspective focuses on **iMarker-related integration challenges**, which are predominantly system- and hardware-oriented. These limitations concern sensor design choices, deployment feasibility in real-world robotic settings, and the robustness of the associated detection algorithms. The second perspective addresses limitations related to the **robotics situational awareness framework** introduced in this thesis. This aspect is algorithmic and conceptual in nature, as it builds upon evolving visual SLAM and graph-based representations, within which iMarkers serve as a supportive rather than central semantic and structural cue. Separating the analysis into these two aspects allows for a clearer distinction between practical integration constraints and higher-level methodological boundaries.

3.1.1 iMarker Integration into Robotics

Despite the demonstrated feasibility of iMarkers in [164, 52], a comprehensive assessment across diverse sensing modalities, robotic platforms, and real-world operating conditions remains an open challenge. The primary reason for this was the ongoing nature of the research project, particularly in iMarker fabrication and production scalability. Thus, evaluations of the potential to integrate iMarkers into robotics applications are limited to a subset of sensor configurations and controlled experimental scenarios. This leaves broader generalization to heterogeneous robotic systems and deployment contexts for future investigation.

One key limitation concerns the design and maturity of the **iMarker fabrication procedure**. Although the development of a fully automated iMarker production pipeline

was outside the scope of the current thesis, it was actively investigated at various stages of the TRANSCEND project. As a result, iMarker fabrication remained partially manual, constraining the number, diversity, and consistency of markers available for experimental evaluation. This limitation directly affected the ability to conduct a comprehensive analysis across a broad range of deployment scenarios. In particular, evaluation under varying illumination changes, diverse weather conditions (mainly for outdoor applications), and long-term environmental influences (*e.g.,* dirt accumulation, surface degradation, or aging effects) could not be thoroughly explored.

Another limitation relates to the **sensor setup** and **detection pipeline** design. In practice, a trade-off exists between iMarker detection robustness and clutter rejection, which varies across sensor types and configurations. In this regard, the choice of sensing modality and associated detection algorithms plays a critical role in system performance, and optimal configurations may differ depending on task requirements and platform constraints. A systematic exploration of this design space has not yet been fully addressed, leaving open the possibility of suggesting a variant of the iMarker detection sensor for different robotics use cases.

A further open challenge relates to the **scope of evaluated robotic applications**. In this thesis, experimental validation was primarily conducted within the *visual SLAM* domain, where iMarkers can encode and provide structured information to support localization and situational awareness. While this represents a well-aligned application scenario, the potential of iMarkers extends beyond VSLAM to other robotic tasks, such as object manipulation, grasping, or task-level reasoning. These application domains were not explored in the current work, mainly due to the substantial time investment and domain-specific expertise required to adapt the iMarker detection sensor and associated perception pipelines to task-specific requirements. Integrating iMarkers into such scenarios would necessitate additional system-level redesign, task-dependent sensing strategies, and extensive experimental validation, which were beyond the scope of this thesis.

Another open challenge concerns the practical justification and **added value of iMarkers** within robotic systems. Although iMarkers are visually imperceptible to humans, aesthetic integration and effective environmental camouflage require additional support for their widespread adoption within the robotics community. To justify replacing conventional printed fiducial markers with iMarkers, it is necessary to thoroughly investigate application scenarios where iMarkers are not merely an alternative but the only *viable solution*. Such scenarios may include environments where visible markers are undesirable, prohibited, undetectable, or impractical due to safety, privacy, or operational constraints. At this stage, these use cases remain only partially explored.

The final open research challenge regarding iMarker integration arises from the requirement for **environment preparation**. Similar to conventional printed markers, iMarkers typically require deliberate placement within the environment to enable reliable operation of the robotic system. In many robotics applications, this prerequisite limits scalability and reduces autonomy, as the system depends on prior environmental modification rather than operating in fully unprepared or unknown settings. Consequently, it is essential to identify scenarios in which the benefits of deploying markers (whether printed or invisible) outweigh the associated costs of installation, maintenance, and environmental intervention. Determining where marker-based solutions are not only effective but also cost-efficient and operationally justified requires further research.

3.1.2 Situational Awareness Framework

In this section, the discussion of limitations and open challenges focuses primarily on vS-Graphs [168]. The primary reason is that the author's earlier frameworks [51, 52] were foundational steps that ultimately led to the development of vS-Graphs into a mature framework. These prior works provided essential insights and design principles, which have since been consolidated and refined within the vS-Graphs system. Additionally, vS-Graphs is an actively evolving framework among the proposed approaches, with ongoing development to improve robustness, expressiveness, and integration into visual SLAM pipelines.

One primary limitation of the vS-Graphs framework is its reliance on a predominantly **GPU-dependent** semantic scene segmentation module. While this design choice enables dense semantic understanding, it limits the applicability of vS-Graphs in resource-constrained settings, such as embedded robotic platforms or low-power autonomous systems without GPU resources. Note that running the CPU-based version of these components yields non-real-time system performance. However, the current implementation of vS-Graphs has not been extensively evaluated under strict computational and memory constraints. Addressing this limitation is particularly important for deploying vS-Graphs in real-world robotic applications that require long-term autonomy and energy efficiency.

Another open challenge concerns the direct dependency of vS-Graphs on the underlying **quality of scene segmentation** module. Since semantic labels play a central role in constructing, optimizing, and updating the scene graph representation, errors in scene segmentation can propagate through the mapping pipeline, thereby negatively affecting 3D reconstruction accuracy and graph consistency. While high-performing segmentation models can improve semantic reliability, their computational demands often pose challenges to real-time constraints. As a result, a fundamental trade-off exists between segmentation accuracy and real-time performance, and identifying segmentation frameworks that offer reliable outputs under strict latency remains an open research issue.

A similar limitation arises from the robustness of semantic segmentation for detecting building components, particularly walls and ground surfaces, which are critical for forming higher-level entities such as rooms. In practice, segmentation models may fail in the presence of appearance variability, such as textured or patterned floors, reflective surfaces, or walls with diverse materials and visual characteristics. These failures can lead to false positives or misclassifications, thereby affecting plane extraction, semantic association, and the accuracy of higher-level abstractions. Although the vS-Graphs framework incorporates a semantic voting mechanism across multiple keyframes to improve reliability, this strategy cannot fully eliminate systematic segmentation errors, especially in visually ambiguous or highly dynamic environments.

3.2 Future Research Directions

Based on the foundations established in this thesis, several research directions emerge to further enhance the capabilities, robustness, and applicability of the proposed robotic perception and situational awareness systems. These directions are motivated not only by the limitations identified in the preceding section but also by the broader potential of the thesis contributions in real-world robotic deployments.

3.2.1 Smartphone-Compatible iMarker Detection

Although the current iMarker detection software framework supports a range of plug-and-play vision sensors, an interesting direction is to develop a lightweight, **smartphone-**

compatible iMarker detector with rapid recognition capabilities. A single-vision iMarker detector mobile application running on standard smartphones would enable iMarkers to be detected without specialized hardware, expanding the technology's applicability for rapid detection and recognition experiments. Given that the existing detection pipeline is already lightweight, a smartphone implementation could serve as a feasible proof of concept for testing iMarker detection strategies in diverse environments. This variant is currently under investigation by a colleague to demonstrate mobile, on-site validation and broaden the accessibility of iMarker-based experiments.

3.2.2 Supporting Additional Semantic Entities

A promising research direction is to extend the vS-Graphs framework by incorporating a broader set of **environment-driven semantic entities** (*e.g.,* building components). Such entities include staircases, ceilings, window frames, doors, and other elements that contribute to a more comprehensive and semantically rich representation of indoor environments. Expanding the set of supported entities would allow vS-Graphs to move beyond basic planar abstractions toward a *higher-level understanding* of scene structure and functionality. While many of these building components can, in principle, be recognized by existing scene understanding and computer vision modules, certain elements (*e.g.,* transparent or reflective structures) remain challenging to detect reliably. In this context, a potential research avenue is the use of **iMarkers** to augment semantic information for **transparent elements**, such as glasses or transparent partitions, while preserving visual imperceptibility for human observers. This approach could provide additional structural cues without introducing visual clutter or altering the environment's appearance.

Enriching the semantic vocabulary in this manner has the potential to significantly enhance digital twin generation, enabling reconstructed environments to more accurately reflect both the geometric layout and functional properties of real-world spaces. Moreover, the inclusion of specific entities, particularly staircases, naturally extends the applicability of vS-Graphs to **multi-floor environments**, where vertical connectivity between levels is critical. Supporting such entities would open the door to more complex real-world scenarios and further strengthen the framework's suitability for long-term autonomous operation in structured indoor environments.

3.2.3 vS-Graphs Scaling

Another direction for future research is the **scaling of vS-Graphs** to larger buildings, multi-floor infrastructures, and collaborative multi-robot mapping scenarios. While the current version of vS-Graphs has demonstrated effectiveness in moderately sized indoor environments, extending it to large-scale, structurally complex spaces poses significant challenges in computational efficiency, memory management, and long-term map consistency. In such scenarios, the number of semantic entities and their interrelations can grow substantially, leading to increased graph complexity and higher optimization overhead. Moreover, large-scale environments pose additional challenges for relocalization, loop closure detection, and coherent hierarchical 3D scene graph representation. To address these issues, strategies such as incremental graph pruning, hierarchical abstraction, or selective optimization may be required to preserve the framework's real-time performance.

Scaling vS-Graphs also naturally aligns with **multi-robot collaborative mapping**, where multiple agents contribute to a shared situational awareness space. In these settings, additional challenges arise in graph merging, semantic alignment across agents, and consistency maintenance under partial observations and asynchronous updates.

3.2.4 Reasoning-based Structural Element Recognition

Another potential direction is the integration of **Graph Neural Network (GNN)-based room detection** and semantic reasoning within the vS-Graphs framework. Given the inherently graph-structured nature of vS-Graphs, GNNs offer a natural mechanism for learning spatial relationships, structural patterns, and higher-level semantics directly from the 3D scene graph representations. By employing *learned representations* of room shapes, layouts, and connectivity patterns, GNN-based models could enable automatic room detection and classification beyond purely geometric heuristics. As shown in [169], this would enable the system to infer room layouts from structural and semantic cues encoded in the graph. Such learning-based reasoning could enhance robustness in complex or large-scale environments by capturing contextual dependencies between walls, doors, and other architectural elements.

3.2.5 Semantic Loop Closure Detection in vS-Graphs

The current version of vS-Graphs lacks an explicit mechanism for **semantic loop closure detection** that operates at the levels of building components and structural elements. While geometric and appearance-based loop closure is employed within the underlying VSLAM pipeline, they do not fully exploit the rich semantic information encoded in vS-Graphs. Incorporating semantic loop closure reasoning would allow the system to detect previously visited places based on structural similarity, semantic layout, or recurring spatial patterns, even under significant viewpoint changes or perceptual aliasing. For instance, recognizing similar room configurations, wall arrangements, or connectivity patterns could enable robust loop closure in environments where purely visual cues are insufficient. This requires addressing several aspects, including appropriate implementations of similarity metrics, handling partial or evolving observations, and ensuring scalability in large-scale environments.

3.2.6 Incorporating Architectural Priors in vS-Graphs

Another promising research direction is to incorporate environmental descriptions via **architectural priors**, such as floor plans and Building Information Models (BIM), into the vS-Graphs framework to enhance mapping accuracy and scene understanding. Using such priors can provide strong structural cues that complement data-driven perception, particularly in environments with sparse visual observations. This contribution can improve semantic consistency, reduce drift, support robust relocalization, and enable higher-level reasoning about spatial layout and connectivity. For instance, BIM models can offer rich geometric and semantic context for validating reconstructed structures, enabling applications such as deviation detection and progress monitoring in construction sites.

These directions are currently being explored within the context of the **informed vS-Graphs (ivS-Graphs)** project [170], which aims to combine graph-based situational awareness with infrastructure-level knowledge. Nevertheless, incorporating architectural priors also introduces challenges, including model alignment, handling outdated or incomplete prior information, and balancing prior constraints with sensor-driven observations.

3.2.7 Exploring Vision-Language Models

Vision–Language Models (VLMs) are robust multimodal AI systems that combine underlying computer vision encoders with Large Language Models (LLMs) to interpret visual inputs, including video frames. By bridging visual perception and linguistic understanding, these visual embeddings can facilitate the provision of constructive environment

descriptions and interaction with the constructed 3D scene graphs. Thus, integrating them within the vS-Graphs framework can enable contextual reasoning and natural-language-based scene interpretation, making vS-Graphs the core module of a wide range of robotics situational awareness frameworks. For instance, it would allow robots to interpret and reason about reconstructed environments using high-level semantic concepts expressed in natural language, such as descriptions of spatial relationships or functional regions encoded within vS-Graphs. This capability can significantly improve explainability, enabling robots to justify decisions or actions by referencing semantic entities and relationships in a human-understandable manner.

3.3 Closing Insights

The thesis in hand explored integrating **semantic scene understanding**, **advanced robotic perception**, and **artificial landmarks** (spanning classic paper-based fiducial markers to novel imperceptible iMarkers) into real-time robotics systems to improve situational awareness and autonomy. The work began with **interdisciplinary collaborations** to develop a new generation of robotics landmarks with minimal human-centered disturbance and extended to practical software frameworks in robotics. The thesis builds on **five peer-reviewed papers** by the author, each addressing a distinct aspect of the project and forming a coherent research trajectory that advances the state of the art.

In this regard, the thesis addresses the limitations of traditional fiducial markers in the robotics community by proposing **iMarkers** that seamlessly blend into their surroundings. Within the **robotics perception** domain, it covers iMarker fabrication considerations, novel detector sensor designs, and the corresponding implemented algorithms. As a case study, the thesis presents a **marker-based Visual SLAM** system for environment representation, in which fiducial markers placed on walls enable the extraction of higher-level entities. This contribution shows that simple geometric cues from markers can be elevated into meaningful semantic structures. To further enrich environmental understanding, the work introduces **hierarchical scene graphs** with keyframes, map points, fiducial markers, walls, corridors, and rooms across multiple layers. This structured design enables the generation of human-interpretable digital twins. The thesis also presented the first **robotics-oriented integration of iMarkers**, demonstrating that they can be successfully detected, assigned to semantic elements, and incorporated into VSLAM pipelines. Although used in limited quantity due to fabrication constraints, the experiments confirmed that iMarkers provide the same **semantic utility** as classical markers, validating their potential for unobtrusive mapping and augmented indoor environments. The final and most significant contribution, **vS-Graphs**, unified the insights from previous work into a more robust, scalable, and marker-optional VSLAM framework. By tightly coupling visual perception, RGB-D Visual SLAM, and semantic scene understanding, it overcomes the dependency on artificial landmarks while still exploiting them as optional sources of semantic enrichment. vS-Graphs contains parallel threads to detect **building components** (walls and ground surfaces) and **structural elements** (rooms and floors) using visual and depth data. By embedding all outputs into an optimizable 3D scene graph, the system provides a consistent representation suitable for navigation, mapping, and environment-aware decision making.

Overall, this thesis demonstrates that robots can achieve strong **situational awareness** through hybrid approaches that blend *geometry*, *semantics*, and *artificial cues* (mainly iMarkers). These findings led to a more adaptive, transparent, and human-aligned environment modeling and digital twin generation for robotics applications.

Part II

Appended Articles

4

iMarker Detection

The first paper [164] was submitted to **IEEE Robotics & Automation Magazine (RA'M)** in March 2025 and is conditionally accepted[1]. This work introduces the concept of *iMarkers* and demonstrates their application in advanced robotics, detailing the fabrication process, sensor design, algorithmic implementation, and experimental evaluation. Furthermore, the proposed detection code has been made publicly available to support reproducibility and further research.

Paper Card

Title: Unveiling the Potential of iMarkers: Invisible Fiducial Markers for Advanced Robotics
Authors: A. Tourani, D.I. Avşar, H. Bavle, J.L. Sanchez-Lopez, J. Lagerwall, and H. Voos
Venue: IEEE Robotics & Automation Magazine (conditional acceptance)
DOI: `https://doi.org/10.48550/arXiv.2501.15505`
Web-page: `https://snt-arg.github.io/iMarkers/`
Repository: `https://github.com/snt-arg/iMarker_detector_standalone`

Copyright

© 2026 by the authors. This article is an open-access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (`https://creativecommons.org/licenses/by/4.0/.`)

[1] A revised version was submitted in January 2026.

4.1 Abstract

Fiducial markers are widely used in robotics for navigation, object recognition, and scene understanding. While offering significant advantages for robots and Augmented Reality (AR) applications, they often disrupt the visual aesthetics of environments, as they are visible to humans, making them unsuitable for many everyday use cases. To address this gap, this paper presents "*iMarkers*," innovative, unobtrusive fiducial markers detectable exclusively by robots and AR devices equipped with adequate sensors and detection algorithms. These markers offer high flexibility in production, enabling customization of their visibility range and encoding algorithms to meet various requirements. The paper also introduces the hardware designs and open-sourced software algorithms developed for detecting iMarkers, highlighting their adaptability and robustness in the detection and recognition stages. Numerous evaluations have demonstrated the effectiveness of iMarkers relative to conventional (printed) and blended fiducial markers and have confirmed their applicability across diverse robotics scenarios.

4.2 Introduction

Fiducial markers, *i.e.,* artificial landmarks with distinguishable graphical patterns, are widely used in robotics and computer vision to offer reliable object identification, localization, and pose estimation [171]. As markers provide better-defined reference points than naturally available ones, they are "*simple-yet-effective*" solutions when consistent feature matching and precise pose retrieval are essential. Additionally, augmenting information with markers and employing appropriate systems to extract it can benefit various domains, including AR and robotics [172]. For instance, markers can narrow the range of objects a robot needs to recognize and provide real-time pose information for tracking and localization. Thus, decoding data from markers detected using the robot's camera is less costly than processing the entire visual scene that the robot can see.

Although they can offer various advantages, observing multiple objects labeled with printed markers can be aesthetically undesirable and disruptive. In other words, many would prefer to avoid seeing printed tags around them and want the robots to operate effectively without spoiling the zones with these artificial landmarks. Evidence also shows that visible markers can disrupt natural gaze behavior [173]. Consequently, and building upon the idea introduced in [10] for using CSRs to create imperceptible landmarks, the authors explored the use of team-developed CSRs and propose a new generation of robotics fiducial markers, dubbed *iMarker*[2]. This is achieved by replacing the printing-ink pigment used in common markers with spherical-shell CSRs. Due to the peculiar behavior of CSRs, iMarkers can be indistinguishable to human eyes but observable by robots if a dedicated sensor is provided. Moreover, such markers can be fabricated to be detected only within specific wavelength ranges to avoid visual clutter. Fig. 4.1 shows that, in qualitative comparisons, iMarkers are hardly noticeable to the naked eye or in standard camera frames, compared with traditional printed fiducial markers.

The motivation of this paper is to demonstrate the role of iMarkers in robotics by presenting their design and detection procedures and benchmarking their applicability in real-world robotic scenarios. Moreover, the work aims to validate the value of iMarkers in simplifying challenging robotic perception tasks, such as detecting transparent and reflective surfaces (*e.g.,* glass, windows, and mirrors). Hence, the leading contributions of the paper are summarized below:

[2] `https://snt-arg.github.io/iMarkers/`

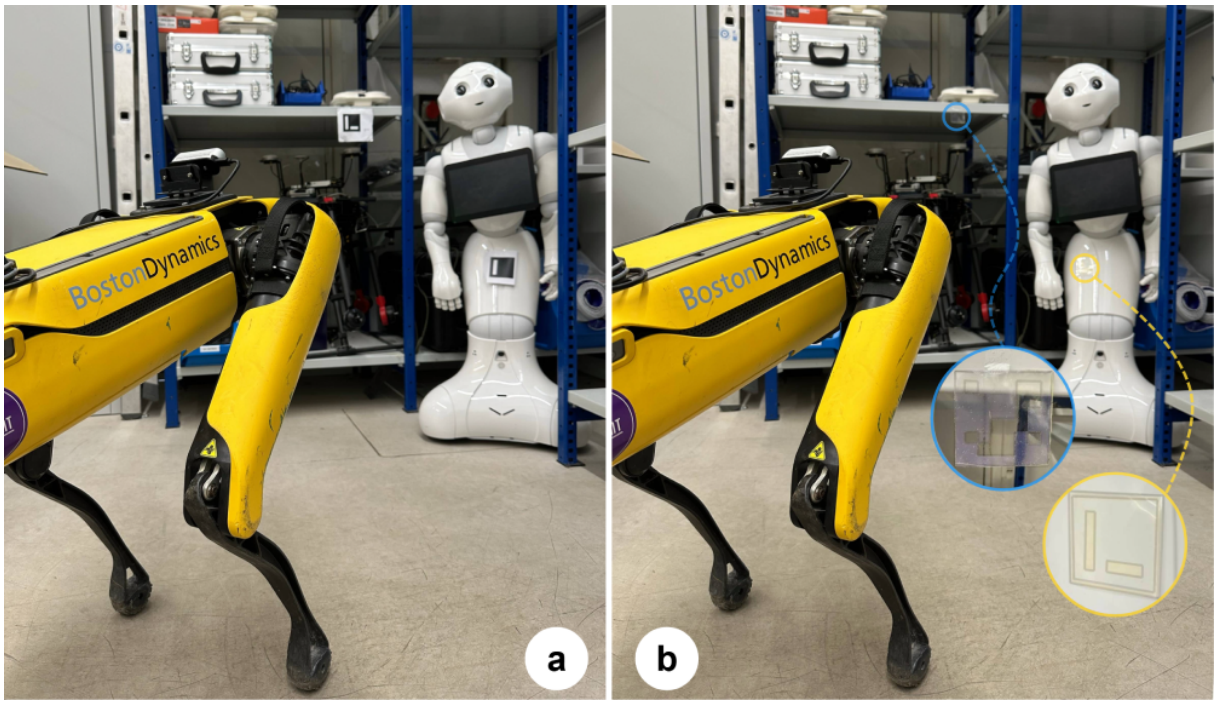


Figure 4.1: A legged robot observing an environment labeled with multiple **a)** printed fiducial markers and **b)** the proposed iMarkers, highlighted and magnified.

- Introducing iMarkers for robotics and AR applications; invisible fiducial markers highly flexible in design and undetectable by the naked eye,
- Proposing multiple sensor solutions and open-source software algorithms for detection and 6DoF pose estimation of iMarkers of various designs; and
- Assessing their applicability relative to traditional fiducial markers, with emphasis on scenarios where conventional printed markers are ineffective.

4.3 Literature Review

4.3.1 Ordinary Fiducial Markers

Fiducial marker libraries are expected to enable rapid detection, reliable recognition, and efficient information decoding. Ordinary markers, mainly printed on paper, can be classified into *non-square* (designed as circles, point sets, or arbitrary visual patterns), *square* (or matrix-based), and *hybrid* variations. In non-square markers, identification and data decoding generally occur *w.r.t.* the center of their geometric shape. InterSense [53] and WhyCode [57] are circular-shaped libraries in which the correspondence point detection is the tricky part. Dot-shaped libraries, *e.g.,* RUNE-Tag [60], provide fast detection but perform poorly in noisy conditions. Other arbitrary, shape-free markers are primarily used in 6-Degrees-of-Freedom (6DoF) pose estimation. Despite robust detection, they face challenges, including inter-marker-encoded dot confusion and steep viewing-angle recognition.

Square fiducial markers, on the other hand, employ binary codes encoded in matrices and deliver four principal corner points for precise pose estimation. They require lower computational cost for recognition, enabling faster detection algorithms. ARToolkit [75] is a matrix-based marker library that uses template-matching for square marker pattern

determination. However, it exhibits high reprojection errors under rotation or camera-angle changes. AprilTag [81] uses gradient-based edge detection and reliable line-segment extraction to identify potential tag borders, ensuring accurate marker recognition even under challenging viewing angles. Its geometrically driven detection pipeline, combined with built-in error-correction mechanisms, significantly reduces false positives caused by background textures and perspective distortions, making AprilTag more robust to rotation, occlusion, and warping than traditional systems. Similarly, Garrido-Jurad *et al.* introduced ArUco Marker [82], which supports configurable marker dictionary generation with maximized inter-marker Hamming distances. It includes automatic marker detection with error correction via dictionary search, as well as a novel occlusion-handling method using color-based masks for robust pose estimation. The ArUco pipeline uses adaptive thresholding to segment high-contrast regions and separate marker candidates, followed by square-shape isolation and perspective transform computation, which extract bit patterns for predefined dictionary matching.

Despite their widespread use and robust performance, these markers, whether non-square or square, remain visually intrusive and can disrupt the natural aesthetics of environments. This limitation has motivated the development of markers that blend more seamlessly into their surroundings.

4.3.2 Blended and Unobtrusive Fiducial Markers

In contrast to paper-based markers, some works propose solutions for scenarios that require specialized detection sensors. In this regard, ArTuga [174] is a multimodal marker for detection in challenging environments through photometric and radiometric sensor fusion. Seedmarker [88] considers the aesthetic properties of markers by embedding them in physical objects, *e.g.,* laser-cut plates and 3D-printed tangibles. ACMarker [90] is another approach that can be detected by custom sensors in underwater robotics applications.

Other approaches focus on integrating less obtrusive markers into environments to minimize their visual impact while maintaining robust detectability. Works in [10, 12] introduced the potential of creating imperceptible visual landmarks by exploiting the optical properties of particular materials. However, they remain primarily focused on materials and optics and do not investigate the applicability of landmarks, such as fiducial markers, for perception robustness, validity, or accuracy. Despite briefly exploring the potential integration of such landmarks in a robotics context in [52], it does not provide a concrete sensing-modality design, a description of detection algorithms, or a systematic experimental evaluation. TRDM [91] uses randomly distributed printed dots on transparent sheets to achieve low visual saliency. Nevertheless, geometric point matching and marker detection are tricky while integrating pairwise relationships among the detected dots. InfraredTags [93] are discreet, durable markers printed and embedded in various objects, providing invisibility and enabling rapid scanning. However, their fabrication is limited to the IR range, reducing design flexibility, and their thin, embedded form makes them difficult to locate visually, complicating alignment of IR illumination and sensing during deployment. BrightMarkers [92] are 3D-printed fluorescent filaments embedded in objects and require NIR cameras for detection. Aircode [95] embeds markers directly into object fabrication, enabling seamless integration for use cases such as robotic grasping. Nevertheless, it suffers from slow decoding times (tens of seconds) that depend on the camera's viewing angle.

Although the surveyed blended and unobtrusive methodologies have demonstrated effectiveness in robotics and AR tasks by keeping markers discreet, their applicability in

Table 4.1: A comparison of the primary features and capabilities of various unobtrusive fiducial marker solutions.

	AirCode [95]	**InfraStructs** [94]	**InfraredTags** [93]	**iMarker** (ours)
Unobtrusiveness	✓	✓	✓	✓
Core material	air pockets	polystyrene	IR filament	CSRs
Versatile pattern coding	✗	✗	✓	✓
Fabrication cost	high	normal	low	low
Fabrication simplicity	✗	✗	✓	✓
Production versatility *	*Vis*, *UV*	*IR*	*IR*	*IR*, *UV*, *Vis*
Sensor cost	very high	high	low	low
Detection time range	*second*	*second*	*millisecond*	*millisecond*

**IR*, *UV*, and *Vis* refer to Infrared, Ultraviolet, and visible ranges, respectively.

real-world scenarios is often constrained by the trade-off between "*invisibility*" and "*detection speed.*" In other words, solutions like AirCode achieve high levels of invisibility but suffer from significantly slow detection speeds. Other crucial factors include simplicity, cost-effectiveness, versatility in detection, robust performance, and flexibility in fabrication across various light spectra. These dimensions significantly influence the adoption of such marker solutions compared with easily printable traditional markers.

4.4 iMarkers: Concept and Comparison

iMarkers offer an innovative, versatile solution to address the gaps mentioned, ensuring transparency, flexibility, and unobtrusiveness. Given appropriate sensors and detection algorithms, they can be fabricated to achieve high contrast for detection while remaining undetectable to the naked eye. Table 4.1 highlights the characteristics of iMarkers compared to other blended methodologies. Accordingly, although all solutions aim to remain hidden from human perception, iMarkers cover distinct aspects that set them apart:

Core Material: The material used to make iMarkers is CSRs [12], microscopic droplets of Cholesteric Liquid Crystals (CLCs) that self-organize into a helical structure, making them highly effective at reflecting specific wavelengths of light, including IR, UV, or visible. Their spherical shape makes them omnidirectional, selective retroreflectors, ensuring detectability regardless of viewing angle. They produce vivid, circularly polarized iridescent colors through selective light reflection, enabling clear separation between CSR and non-CSR regions. As shown in Fig. 4.2, iMarkers combine CSRs for the marker patterns with transparent UV-curable glue to maintain the structure of the fiducial marker. An alternative fabrication method involves directly spraying CSR shells onto surfaces, a promising approach currently under investigation.

Pattern Coding and Recognition: iMarkers support various coding patterns for recognition and information retrieval. This paper focuses on ArUco-based [82] markers with variant dictionaries, but other standard fiducial marker libraries can also be adapted to the iMarker design.

Fabrication Cost and Efficiency: The cost and simplicity of fabrication significantly impact the widespread adoption of markers. While AirCode requires partitioning and InfraStructs depend on a complex, layered assembly, iMarkers, as shown in Fig. 4.2, can be fabricated using cost-effective methods such as dispensing or spray coating.

Production Versatility: In contrast with other solutions, iMarkers offer production flexibility, enabling design and detection across the entire light spectrum. As shown in

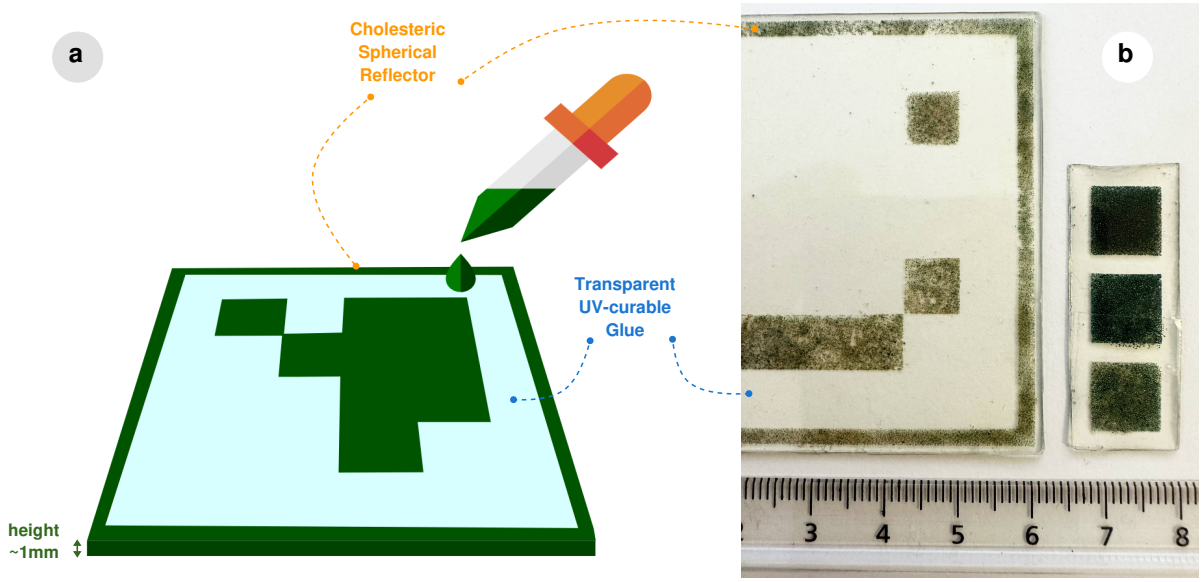


Figure 4.2: The role of CSRs in the design of iMarkers: **(a)** an iMarker featuring ArUco patterns filled with CSRs, **(b)** an iMarker displayed next to visible-range CSR particles, illustrating various shades of green for fabrication.

Fig. 4.3, UV-range iMarkers remain entirely transparent and invisible to the naked eye, even when placed on a transparent surface such as glass. IR-range iMarkers are similarly discreet, though slight scattering or subtle color effects may reveal their presence (to some extent). Even in the visible range, iMarkers can be designed to blend with patterned backgrounds or camouflage against similarly colored surfaces, ensuring minimal visual intrusion.

Sensor Affordability: Detection requirements vary across unobtrusive marker systems: AirCode relies on projector-camera setups, InfraStructs require costly Terahertz scanners, and InfraredTag needs specialized imaging modules with microprocessors. In contrast, iMarkers can be detected using affordable sensors equipped with standard optical components, provided that the camera is compatible with the polarization and wavelength characteristics of the CSR material.

Detection Algorithm Performance: Since the sensors are designed to generate binary images, fiducial marker patterns are detected within milliseconds, ensuring real-time performance for various robotics and AR use cases.

4.5 Sensor Design and Detection Strategies

Three primary solutions are proposed for iMarker detection, including "*dual-vision,*" "*dynamic single-vision,*" and "*static single-vision,*" further described in the following:

4.5.1 Dual-vision Solution

Sensor Design. The first setup is a homogeneous perception system comprising two synchronized, perpendicularly positioned cameras, each facing a different surface of an optical beamsplitter. As shown in Fig. 4.4, the sensor can be assembled with either a cube or a plate beamsplitter. The beamsplitter equally divides the incoming light (*i.e.,* 50:50) into two parts: the transmitted and the reflected light. By mounting the cameras and the beamsplitter in a cage, one camera captures the scene using the transmitted light, while the other captures it using the reflected light, making two perspectives with



Figure 4.3: Versatility in iMarker production: **(a)** a 6×6 cm^2 UV-range iMarker, **(b)** an IR-range iMarker with the same dimensions, **(c)** the UV-range iMarker placed on a transparent surface, **(d)** a 7×7 cm^2 dark green visible-range iMarker camouflaged with its background.

minimal variation. Attaching identical circular polarizers to each camera's lens effectively blocks light with the opposite circular polarization, ensuring only the desired handedness is detected. Due to the inversion of circular polarization upon reflection at the beamsplitter, the camera capturing the reflected light detects the "opposite" circular polarization to that of the camera capturing the transmitted light. Hence, the setup simultaneously captures the same scene with two cameras, ensuring that the CSR regions of iMarkers in the sensor's FoV appear blocked in one camera while remaining visible in the other (due to the opposing circular polarization of the CSR reflections).

Note that the cameras in this setup must be synchronized to capture the same scene simultaneously (frame $f_t^{C_1}$ captured by camera C_1 at time t should depict the same scene as frame $f_t^{C_2}$ of camera C_2). As the cameras' frame rate and shutter speed are vital for high-

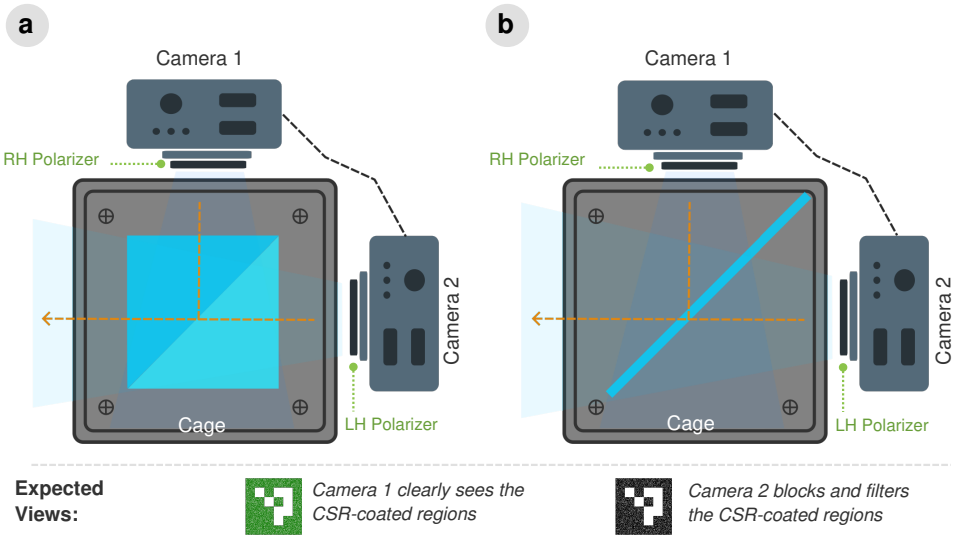


Figure 4.4: Dual-vision iMarker detection setups using synchronized cameras with circular polarizers and either a cube **(a)** or plate **(b)** beamsplitter.

speed robotic applications (*e.g.,* drone navigation), integrating global-shutter cameras in this sensor is highly recommended. The calibration procedure involves capturing a standard calibration pattern (*e.g.,* a chessboard) by each camera. The camera's intrinsic (*e.g.,* focal length and distortion coefficients) and extrinsic parameters are required for iMarker pose estimation.

Detection Algorithm. Since the non-CSR regions of the scene lack circular polarization and appear identical in both cameras, simple frame subtraction can detect iMarkers with high contrast, as shown in Algorithm 1. Accordingly, the iMarker detection procedure requires "*spatial subtraction*" of the synchronized frames captured by each camera. Having $\mathbf{F}_{C_1}$ and $\mathbf{F}_{C_2}$ as the synchronized frame sets captured by calibrated cameras C_1 and C_2, each frame $f_1 \in \mathbf{F}_{C_1}$ finds its corresponding synchronized frame $f_2 \in \mathbf{F}_{C_2}$. The next stage is to align f_2 with respect to f_1 using the alignment parameters h (captured during calibration), where h is the homography matrix computed from matching Oriented FAST and Rotated BRIEF (ORB) features. Subtracting the "aligned" f_2 from the "original" f_1 frame results in the final binary image f_p containing the input images' dissimilar parts. Since the CSR region of iMarkers is visible in one frame and blocked in the other, only the patterns created by CSRs are expected to remain in f_p. Then, a thresholding process with a value of θ, followed by post-processing, is applied to f_p to generate a binary frame that enhances the iMarker patterns' visibility. The final processed frame contains potential iMarkers $\mathbf{M}$ at time t.

4.5.2 Dynamic Single-vision Setup

Sensor Design. The second approach offers a single camera equipped with a switchable (dynamic) circular polarizer alternating between left- and right-handed polarization (Fig. 4.5a), which will be described in detail in a forthcoming paper. In brief, it comprises a fixed circular polarizer paired with a nematic liquid crystal film that functions as a $\lambda/2$-plate in its relaxed (OFF) state and loses its optical functionality when an electric field is applied (ON). A $\lambda/2$-plate reverses circular polarization; therefore, by switching the liquid crystal film between its ON and OFF states (controlled by a function generator synchronized with the camera's frame rate), the camera sequentially captures frames with

Algorithm 1 Dual-vision iMarker detection.

Require: frame-sets F_1 and F_2 from cameras C_1 and C_2
Ensure: list of detected fiducial markers M

```
M ← []
p1 ← calibrate C1
p2 ← calibrate C2
h ← align (F1, F2) considering p1 and p2
while f1 in F1 do
    f2 ← corresponding sync frame in F2
    f2 ← align f2 based on f1 using h
    fp ← f2 − f1 {final subtracted image}
    fp ← apply thresholding on fp
    fp ← post-process fp {erosion + Gaussian blur}
    if marker m found in fp then
        add m to M
    end if
end while
return M
```

right- and left-handed polarization. In an ideal setup, camera frame f_t captures the scene with a particular polarization (left- or right-handed), and frame f_{t+1} captures it with the opposite polarization. Thus, the function generator should send signal $\mathbf{S}_g$ to trigger the liquid crystal film at time t as:

$$\mathbf{S_g} = \begin{cases} \text{on}, & \text{for } t = 1, 3, \ldots, 2n-1, \\ \text{off}, & \text{for } t = 2, 4, \ldots, 2n. \end{cases} \tag{4.1}$$

The liquid crystal cell's OFF-to-ON switching time is negligible, but the ON-to-OFF relaxation is slower. Thus, intermediate frames with undefined polarization at high frame rates may need to be discarded during iMarker detection. Once optimized for the liquid crystal cell, it ensures that CSR-coated regions in the iMarkers are blocked in one processed frame and visible in the next.

Detection Algorithm. Inspired by the "*spatial*" frame processing in the dual-vision setup, this approach employs a frame-level algorithm that uses "*temporal*" subtraction to detect the iMarker patterns. Assuming the liquid crystal film operates sufficiently fast *w.r.t* the video frame rate to produce sequential frames with alternating polarization states (Equation 4.1), the consecutive (temporal) frame subtraction algorithm to detect iMarkers using a switchable polarizer is shown in Algorithm 2. Here, each frame ${f_{t-1}}^{\text{off}}$ and its subsequent frame ${f_t}^{\text{on}}$, captured with alternating polarization states, are subtracted to produce a grayscale image f_p, which can then be enhanced through post-processing to facilitate iMarker detection. Additionally, the current frame is stored for subtraction from its succeeding frame in the next iteration. As with the previous approaches, the final stage involves detecting markers $\mathbf{M}$ within the frame f_p.

4.5.3 Static Single-vision Setup

Sensor Design. Another configuration offers a solution for iMarker detection using a single camera equipped with a static circular polarizer attached to its lens (Fig. 4.5b). The polarizer is selected to block the circular polarization reflected by the CSRs forming the

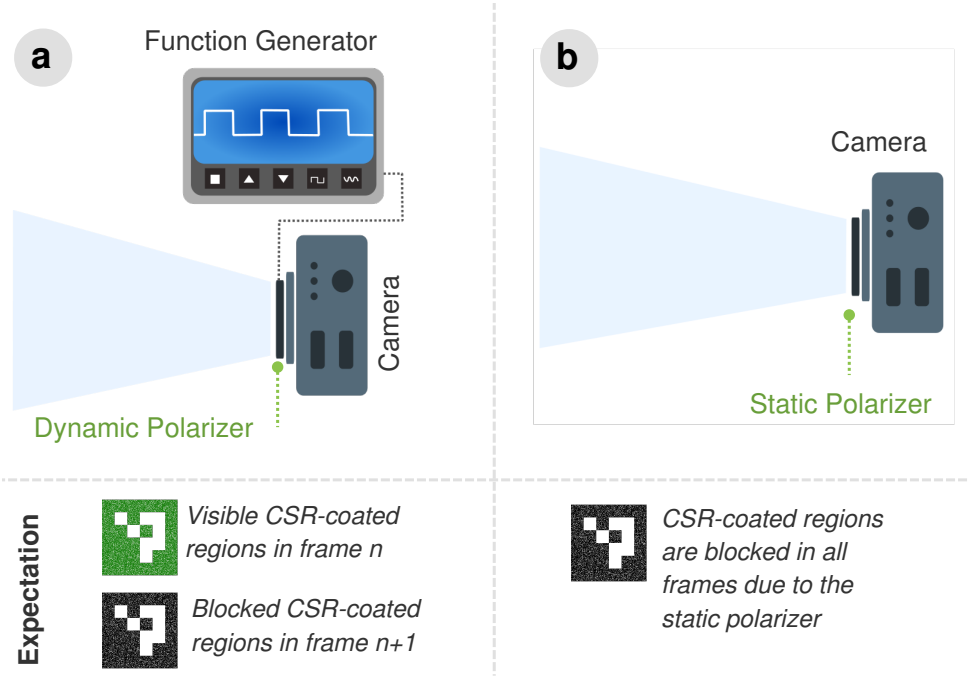


Figure 4.5: Single-vision setup variants designed for iMarker detection, each containing a camera with attached dynamic **(a)** or static **(b)** polarizers.

iMarker pattern. For instance, a left-handed circular polarizer will block iMarker regions with CSRs reflecting right-handed polarization, and vice versa. Thus, a visible-range iMarker with CSRs reflecting right-hand polarized light on a same-color background (*e.g.,* green), indistinguishable to the human eye or standard cameras, becomes detectable as the sensor's left-handed polarizer blocks the CSR regions, revealing the camouflaged iMarker patterns. This design is well-suited to detecting visible-range iMarkers camouflaged in the environment, and it can be used for IR/UV-range iMarkers with appropriate cameras.

Detection Algorithm. Unlike the dual-vision solution, the algorithm for the static single-vision setup does not eliminate the background. However, it simplifies the integration of an iMarker detection sensor by shifting more computation to the detector, achieving a practical balance between complexity and performance. For the visible-range camouflaged iMarkers, the procedure is shown in Algorithm 3. Accordingly, detecting iMarkers $\mathbf{M}$ in a frame $f \in \mathbf{F}$ involves *RGB* to *HSV* color space conversion to isolate the color range corresponding to the marker. The resulting mask frame f_p highlights the target color range c_r in black, with all other areas shown in white. With the assistance of the polarizer, the mask frame f_p highlights the blocked CSR regions of an iMarker with color c (*e.g.,* green) positioned against a background of the same color. Subsequent stages apply post-processing to eliminate noise, followed by detecting marker patterns $\mathbf{M}$ within f_p. Detecting UV/IR-range iMarkers follows the same stages as the visible-range detection, provided that a UV/IR-range camera is used. However, "*color masking*" is unnecessary because the input is grayscale, and "*range thresholding*" is used to identify intensity variations in CSR regions, thereby highlighting the iMarker patterns, as introduced in Algorithm 4.

Algorithm 2 Dynamic single-vision iMarker detection.

Require: frame-set F from the camera
Ensure: list of detected fiducial markers M

```
M ← []
f_prev ← null
while f_t in F do
    f_p ← f_t − f_prev {final subtracted image}
    f_p ← post-process f_p {erosion + Gaussian blur}
    f_prev ← f_t
    if marker m found in f_p then
        add m to M
    end if
end while
return M
```

Algorithm 3 Static single-vision iMarker detection (masking).

Require: frame-set F from the camera (RGB)
Ensure: list of detected fiducial markers M

```
M ← []
r ← (low, high) {demanded color range in HSV}
while f in F do
    f_p ← convert f to color space HSV
    f_p ← filter f_p with the range r
    f_p ← post-process f_p {erosion + Gaussian blur}
    if marker m found in f_p then
        add m to M
    end if
end while
return M
```

4.6 Benchmarking and Evaluation

4.6.1 Operational Implementation

Sensor. Fig. 4.6 depicts the different sensor configurations developed for iMarker detection. The dual-vision setup shown in Fig. 4.6a comprises two ELP 8MP HD camera modules with 75° FoV facing a Thorlabs polarizing cube beamsplitter with a 50:50 split ratio, all mounted in a compact cage assembly. The other dual-vision setup in Fig. 4.6b comprises two synchronized U3-3271LE iDS cameras, aligned with a Thorlabs polarizing plate beamsplitter, housed within a custom-designed 3D-printed cage. The static single-vision setup in Fig. 4.6c uses an Intel® RealSense™ D435 camera with a left-handed circular polarizer affixed to its RGB lens, effectively blocking the right-handed CSR-coated areas of camouflaged iMarkers. Finally, the dynamic single-vision sensor setup in Fig. 4.6d incorporates a switchable polarizer controlled by a function generator mounted on an iPhone 13 case and aligned with its camera. While this prototype relies on manual polarizer switching, we are investigating a version in which the detection software dynamically synchronizes the camera and the function generator to adjust the polarization.

Algorithm 4 Static single-vision iMarker detection (thresholding).

Require: frame-set F from the camera ($grayscale$)
Ensure: list of detected fiducial markers M
  $M \leftarrow []$
  $r \leftarrow (low, high)$ {demanded gray-level range in $grayscale$}
  while f in F **do**
    $f_p \leftarrow$ filter f_p with the range r
    $f_p \leftarrow$ post-process f_p {erosion + Gaussian blur}
    if marker m found in f_p **then**
      add m to M
    end if
  end while
  return M

Software. To support the detection algorithms outlined in Section 4.5, an open-sourced[3] iMarker detection software framework has been developed. The framework is implemented in Python and features a ROS2 interface, enabling straightforward integration into robotic applications.

4.6.2 Qualitative Evaluation

Given the complexity of measuring aesthetic perception, this section aims to **visually assess** iMarkers and showcase their detection performance. To achieve this, various iMarkers were strategically placed in fixed real-world locations and captured by multiple sensors, highlighting their seamless integration across diverse settings. While aesthetic perception is inherently subjective, the visual comparisons in Fig. 4.1 and Fig. 4.3 provide practical evidence supporting the unobtrusiveness of iMarkers across diverse environments.

Fig. 4.7 depicts several qualitative results, showing the versatility of iMarkers across different spectral ranges and detection setups. Accordingly, Fig. 4.7a shows a green iMarker in the visible range discreetly embedded on a similarly colored background, demonstrating its visual camouflage and unobtrusiveness. In this scenario, the static single-vision setup was employed to capture the scene, with the setup's polarizer effectively blocking the CSR regions that form the inner iMarker pattern, thereby enhancing contrast and improving color separation for detection. The detection and pose estimation outcome was obtained using Algorithm 3. Fig. 4.7b shows the output of the dynamic single-vision setup applied to a visible-range iMarker under indoor conditions with controlled illumination. In contrast, Fig. 4.7c presents the corresponding outdoor results, where varying sunlight, shadows, reflections, and background clutter introduce additional challenges for reliable iMarker detection. Notably, the system properly isolates the marker region with high contrast, enabling reliable pattern recognition and data decoding. In both dynamic single-vision scenarios, detection is achieved through temporal subtraction between frames f_{t-1}^{off} and $f_t{}^{\text{on}}$ following the procedure described in Algorithm 2. Fig. 4.7d presents the output of the dual-vision setup applied to a visible-range iMarker placed on a patterned surface under low-illumination conditions. In this configuration, both cameras simultaneously capture the same scene, with the left camera acquiring frame f_l through a left-handed polarizer and the right camera capturing frame f_r through a right-handed polarizer. Marker detection was performed using Algorithm 1, which applies spatial sub-

[3] `https://snt-arg.github.io/iMarkers/`



Figure 4.6: Variant iMarker detection sensors, including the dual-vision setup with cube **(a)** and plate **(b)** beamsplitters, and single-vision setups with static **(c)** and dynamic **(d)** polarizers.

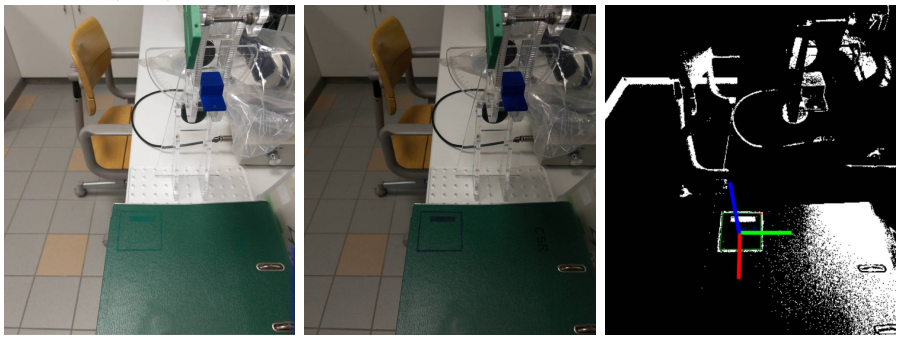
traction between the polarized image pairs. The system effectively suppresses background texture and illumination artifacts, preserving high contrast and reliable detection under challenging lighting conditions. As illustrated in Fig. 4.7e, the UV-range iMarker placed on a transparent surface is virtually invisible to the naked eye. To reveal the inner patterns of the iMarker, near-UV illumination was applied, and the same scene was captured with a UV-range camera, as shown in the middle sub-figure. The method introduced in Algorithm 4 facilitates marker detection by applying a tailored thresholding strategy to the UV image, enabling reliable extraction of iMarker's invisible patterns.

4.6.3 Detection Range and Pose Accuracy Evaluation

To evaluate the effectiveness of iMarkers compared with their paper-printed counterparts, a series of comparative experiments was conducted using both marker variants in the ArUco format. The experiments focused on two primary objectives: (1) determining the maximum detection range across varying viewing angles, and (2) assessing the accuracy of the estimated 6DoF pose for each marker type; both were done using the ArUco detection



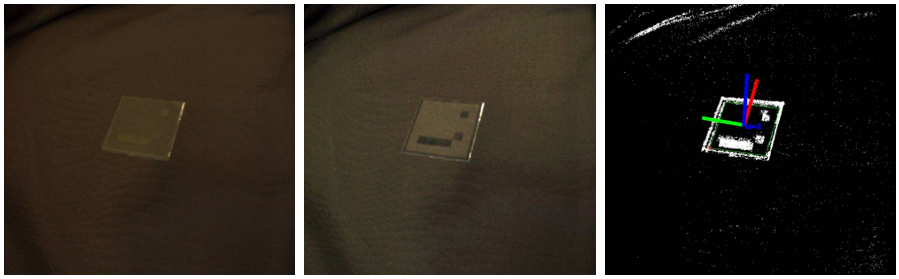
(a) Static single-vision setup detecting a visible-range iMarker: raw frame (left), polarizer applied (middle), and detection result using color masking (right).



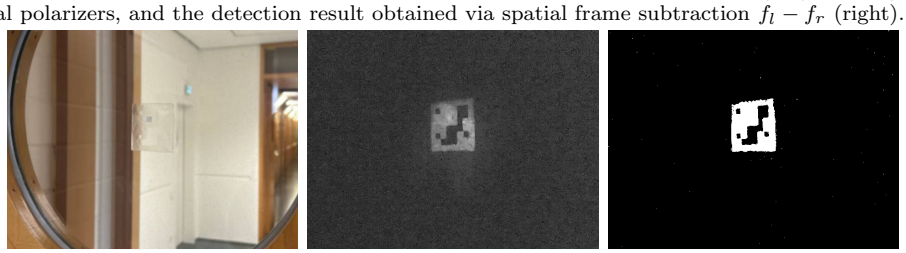
(b) Dynamic single-vision setup detecting a visible-range iMarker: frames f_{t-1}^{off} (left) and f_t^{on} (middle) captured at different times with alternating polarizer states, and the detection result obtained via temporal subtraction (right).



(c) Dynamic single-vision setup detecting a visible-range iMarker in an outdoor environment. The order of the frames (off-polarized, on-polarized, and subtraction) follows the same structure as (b).



(d) Dual-vision setup capturing a visible-range iMarker under low-light conditions: frames f_l (left) and f_r (middle) acquired using orthogonal polarizers, and the detection result obtained via spatial frame subtraction $f_l - f_r$ (right).



(e) Single-vision UV iMarker detection: scene as captured by a standard mobile phone camera (left), corresponding view captured with a UV-sensitive camera under near-UV illumination (middle), and the resulting marker pattern revealing output via thresholding (right).

Figure 4.7: iMarker variants detected using different sensor setups and corresponding algorithms, including the visible- and UV-range iMarkers.

Table 4.2: Mean pose estimation error (in meters) of 7×7 cm^2 markers across different viewing angles. Results are reported for "printed markers" and "iMarkers" (detected using static single-vision (SSV) and dynamic single-vision (DSV) setups) of the ArUco library. Values in brackets indicate the iMarkers' relative increase in error compared to the printed markers at each angle.

	Marker Variants		
Angle	*Printed*	*iMarker (SSV)*	*iMarker (DSV)*
0 °	0.155	0.160 [+3.2%]	0.161 [+3.9%]
15 °	0.191	0.196 [+2.6%]	0.194 [+1.6%]
30 °	0.208	0.211 [+1.4%]	0.214 [+2.9%]
45 °	0.214	0.218 [+1.9%]	0.220 [+2.8%]
60 °	0.228	0.231 [+1.3%]	0.234 [+2.6%]
75 °	0.235	0.238 [+1.3%]	0.241 [+2.6%]

algorithm. The employed sensor across all scenarios was the single-vision setup with the RealSense, unmodified for printed markers and equipped with static/dynamic polarizers for iMarker detection. We intentionally limited the analysis to this setup to ensure a fair comparison for detecting markers, whereas the other sensors in Fig. 4.6 differ significantly in resolution, shutter type, and optical characteristics. All experiments were conducted using the camera's RGB sensor at a resolution of 1024×768 to maintain uniform input quality across evaluations. Regarding marker dimensions, 7 cm $\times$ 7 cm visible-range iMarker variants were used to be compared against 6 cm $\times$ 6 cm and 7 cm $\times$ 7 cm printed ArUco marker references. Evaluations were performed under normal lighting conditions in a controlled robotics laboratory equipped with an OptiTrack motion capture system, providing sub-millimeter ground-truth accuracy for 6DoF pose tracking. Both markers were affixed to a tripod using an adjustable mounting plate, enabling precise control of their orientation relative to the camera. To ensure accurate ground-truth acquisition during the evaluations, omnireflective motion-capture landmarks were placed on both fiducial markers, and their poses were continuously tracked using the OptiTrack system. The examined viewing angle ranged from 0° to 75°, within which the estimated poses of the markers remained reliable *w.r.t.* the ground-truth.

Maximum detection range. According to the evaluation results in Fig. 4.7, increasing the viewing angle reduces the maximum marker-detection range across all scenarios, primarily due to perspective distortion. Additionally, iMarker detection using static and dynamic single-vision setups shows negligible differences in maximum detection ranges, differing between 3 cm to 7 cm, with an average of 4.82 cm. Regardless of the detection setup, printed markers consistently achieved higher detection ranges than iMarkers of identical dimensions (7 cm $\times$ 7 cm). The detection range of 7 cm $\times$ 7 cm iMarkers closely matches that of 6 cm $\times$ 6 cm printed ArUco markers, with differences ranging from 4.12 cm to 6.39 cm and averaging 4.24 cm across various angles of view. It should be noted that printed ArUco markers, thanks to their high-contrast black-and-white patterns, offer substantial visual features that facilitate detection. However, iMarkers demonstrate comparable detection performance in real-world experiments, apart from their distinct advantage over classic markers: their ability to seamlessly blend into the environment without disrupting visual aesthetics. This makes iMarkers highly suitable for shorter-range applications where visual discretion is essential, such as indoor navigation.

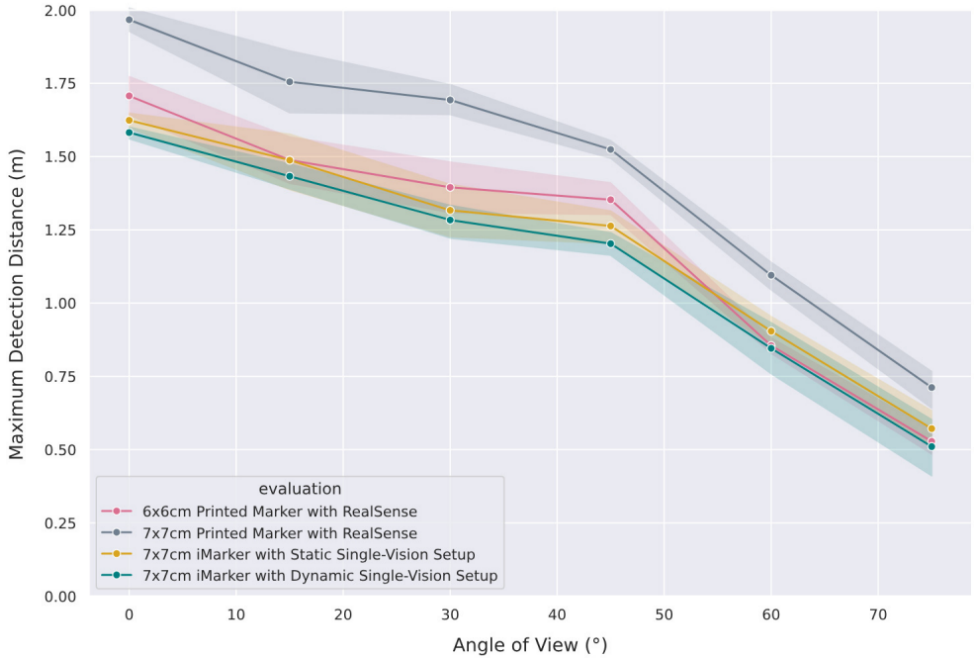


Figure 4.8: Maximum detection distances for "printed" and "iMarkers" (ArUco patterns) at various viewing angles. The shaded regions (error bands) indicate measurement uncertainty in maximum recognition distance.

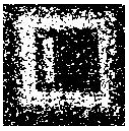
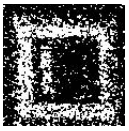
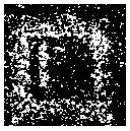
Pose accuracy. The pose estimation accuracy was assessed by measuring the deviation between the detected marker poses and the ground-truth at specified viewing angles. As shown in Table 4.2, the mean pose estimation error norm across various viewing angles remains consistent among the printed ArUco markers and their iMarker counterparts. In general, the detection error increases as the viewing angle grows from 0° to 75°due to increasing perspective distortion of the marker appearance. The relative difference in pose estimation error between printed markers and iMarkers remains within a narrow range of 1.3% to 3.9%. This consistency and negligible differences in error were expected outcomes of our system design, as the proposed hardware-software configurations are explicitly designed to produce pre-processed binary images that reveal the marker patterns before the *detection and pose estimation* stage in the standard ArUco detector. Thus, the marker detector library recognizes both markers by applying built-in preprocessing to either standard RGB inputs of printed markers or the binary outputs generated by the iMarker detector.

Results analysis. It should be noted that the goal of iMarkers is not to outperform printed markers in all performance metrics, but to provide an imperceptible yet reliably detectable alternative. Traditional printed markers benefit from high contrast patterns (*e.g.,* black and white) and inherently offer superior detectability than all blended/unobtrusive systems reviewed in Section 4.3.2. While the invisibility of iMarkers introduces trade-offs in detection robustness, our experimental evaluations of *maximum detection range* and *pose accuracy* show that iMarkers can still deliver competitive results, while maintaining minimal visual intrusion.

4.6.4 Operation in Degraded Visual Conditions

To further highlight the complementary nature of iMarkers relative to conventional markers, an additional evaluation examines scenarios in which iMarkers remain detectable while

Table 4.3: Detection robustness of printed ArUco markers ($\mathcal{P}$) and IR-range iMarkers ($\mathcal{I}$) under progressively lower illumination conditions, highlighting scenarios where conventional markers fail but iMarkers remain reliably detectable.

		Illuminance			
		15 lux	10 lux	5 lux	≤ 5 lux
Sample View	$\mathcal{P}$				
	$\mathcal{I}$				
Sample Detection	$\mathcal{P}$				
	$\mathcal{I}$				
Detection Rate	$\mathcal{P}$	0.72 (13 / 18)	0.50 (9 / 18)	0.11 (2 / 18)	0.06 (1 / 18)
	$\mathcal{I}$	**0.89** (16 / 18)	**0.83** (15 / 18)	**0.83** (15 / 18)	**0.78** (14 / 18)
# False Positive	$\mathcal{P}$	0.17 (3 / 18)	0.33 (6 / 18)	0.33 (6 / 18)	0.44 (8 / 18)
	$\mathcal{I}$	**0.06** (1 / 18)	**0.11** (2 / 18)	**0.17** (3 / 18)	**0.17** (3 / 18)

printed markers fail to be detected. These experiments target degraded visual conditions, including low-illumination and near-dark environments, where visible-spectrum contrast is insufficient for reliable detection of printed markers. To assess real-world applicability in robotics, a representative UAV-landing–inspired marker-detection and positioning task was evaluated under varying low-light conditions. This protocol reflects realistic operational scenarios, such as warehouse automation and night-time inspection, in which robots need to operate reliably under severely degraded visible light.

The comparison is conducted between an IR-range iMarker and a printed ArUco marker of identical physical size (6 cm $\times$ 6 cm), both placed at a distance of 1 m from the camera. Markers use the same ArUco dictionary (original) and differ only in their `marker_id` to support flexible configurations. Ambient illuminance was gradually reduced during the experiment by controlling the environmental light sources, with illuminance levels measured using a calibrated lux meter application. Furthermore, frames captured at matched resolution using the RGB sensor and the IR-range camera with active IR illumination (Ultrafire A100-IR LED torch) and a long-pass filter, with 18 synchronized frames sampled per illumination level for both RGB and IR streams across all scenarios. Hence, the RGB stream is used for printed marker detection, and the infrared stream for iMarker detection.

As summarized in Table 4.3, various complementary metrics, including detection rate

Table 4.4: Performance analysis of iMarkers detection and recognition, based on cumulative time profiling of required modules (averaged over 40 frames) in milliseconds (ms). All Abbreviations are detailed below the table. The input frames are captured at 25fps.

	iMarker Variant				
	UV	*Visible-range*			
Software Module	**SV Static**	**SV Static**	**SV Dynamic**	**DV (cube BS)**	**DV (plate BS)**
Acquisition	11 ms	15 ms	13 ms	32 ms	36 ms
Preprocessing	2 ms	7 ms	9 ms	14 ms	19 ms
Processing	3 ms	23 ms	29 ms	39 ms	42 ms
Post-processing	3 ms	9 ms	8 ms	4 ms	5 ms
Recognition	2 ms	2 ms	3 ms	2 ms	3 ms
Sum	21 ms	56 ms	62 ms	91 ms	105 ms

*PS. *SV*: Single-Vision, *DV*: Dual-Vision, *UV*: Ultraviolet, *BS*: Beamsplitter

and false positive frequency, have been analyzed for printed markers and IR-range iMarkers under progressively decreasing illumination conditions. Here, the detection rate is defined as the ratio of frames in which the correct `marker_id` is detected to the total number of frames captured, all at illumination level l. False positives are incorrect detections triggered by high-contrast background textures that resemble marker patterns, such as repetitive grids or net-like structures.

According to the table, the *detection rate* of printed markers degrades sharply as illumination decreases, because reduced contrast causes marker features to blend into background noise. Specifically, performance drops from 0.72 at 15 *lux* to 0.06 at $\leq$ 5 lux. In contrast, IR iMarkers maintain a largely stable detection rate, varying only slightly from 0.89 to 0.78, with minor degradation attributed to sensor noise and IR signal attenuation under extremely low-light conditions. A similar trend is observed for *false positives*: as illumination decreases, erroneous detections for printed markers increase substantially (from 0.17 to 0.44), primarily due to the reduced *signal-to-noise ratio* in the RGB stream. In contrast, iMarkers maintain a consistently low false-positive rate across all illumination levels, with only minor fluctuations, owing to the CSRs' distinctive IR reflectance properties.

These results show that while printed markers are strongly affected by illumination-dependent failure modes, iMarkers provide reliable detection performance under degraded visual conditions. Additionally, it clearly demonstrates that iMarkers go beyond *aesthetic considerations* and act as an *enabling technology* for robotic perception in scenarios where conventional fiducial markers become unreliable or unusable.

4.6.5 Recognition Speed

Given the need for fast recognition to demonstrate the practicality of fiducial markers for real-time applications (in our context, with an average processing rate of 25 ± 5fps), this subsection presents a detailed profiling analysis of iMarker detection and recognition. Table 4.4 summarizes cumulative processing times for iMarker data decoding. Evaluations were conducted using various sensor configurations and implemented software introduced in subsection 4.6.1. The algorithms developed for all sensor variants share five core mod-

ules: "*acquisition*," which captures video frames; "*preprocessing*," which applies image enhancement techniques; "*processing*," which refines frames to emphasize marker patterns; "*post-processing*," which enhances the final binary images; and "*recognition*" to detect and interpret marker patterns. To obtain the measurements, 7 cm $\times$ 7 cm iMarkers were placed approximately 50 cm from the lens, directly facing the sensor under standard lighting conditions with evenly diffused artificial illumination. For the UV-range iMarker, a DARKBEAM A300 365 nm flashlight was used to enhance the iMarker visibility.

According to Table 4.4, the overall iMarker recognition process, from video frame "*acquisition*" to inner pattern "*recognition*" steps, is completed within milliseconds and adds only $\approx 100\ ms$ overhead to the ArUco library. The reported results were obtained using standard CPU-based processing without the aid of GPUs or other hardware accelerators. For comparison, some methods discussed in Table 4.1 require several seconds for detection, underscoring the efficiency of the iMarker system. Among the various configurations, UV-range iMarker recognition (Algorithm 4) is the fastest, as the input image is already in grayscale, requiring only a simple "thresholding" operation to reveal the marker pattern for recognition. In dual-vision setups (Algorithm 1), "*acquisition*" and "*processing*" account for a significant portion of the processing time, as they require receiving synchronized frames from two cameras and performing image alignment for further processing, respectively. The "*post-processing*" stage is executed more efficiently because the substantial filtering process retains only the regions that differ, *i.e.,* the CSR areas. On the other hand, single-vision setups are faster during "*acquisition*" because they capture frames from a single camera. However, they allocate more time to "*post-processing*," as the captured images exhibit higher noise levels and require refinement of iMarker regions.

4.7 Discussions

4.7.1 Comparative Insights

Fig. 4.9 depicts relative performance scores of different iMarker fabrication variants and the designed sensor configurations for their detection, where larger areas indicate higher values. The relative performance scores were assigned based on theoretical analysis and experimental observations, which will be described in detail.

In Fig. 4.9a, three primary performance aspects of the iMarker variants have been analyzed. Regarding "invisibility," UV-range iMarkers remain entirely imperceptible to the eye, while IR-range iMarkers may exhibit slight visibility due to scattering under extreme lighting conditions. Visible-range iMarkers, on the other hand, are designed to blend into patterned backgrounds; however, upon close inspection, they remain minimally visible to the human eye. When "readability" is studied, visible-range iMarkers are more compatible with standard cameras and do not require additional illumination. However, IR- and UV-range iMarkers require dedicated illumination sources and specific sensors, and their detection performance varies with environmental conditions. Finally, and in terms of "detection equipment cost" (camera, flashlight, *etc.*), visible-range iMarkers are the most economical, as they can be detected using standard cameras. UV equipment (camera and illumination sources) is generally the most expensive of the three, and IR-range iMarker detection equipment costs lie between them.

Considering the sensor-related factors shown in Fig. 4.9b, the performance scores can be studied in five aspects. In design complexity, the static single-vision setup is the most straightforward, requiring only a static polarizer attached to the lens. The dynamic single-vision setup has moderate complexity, incorporating a function generator to control

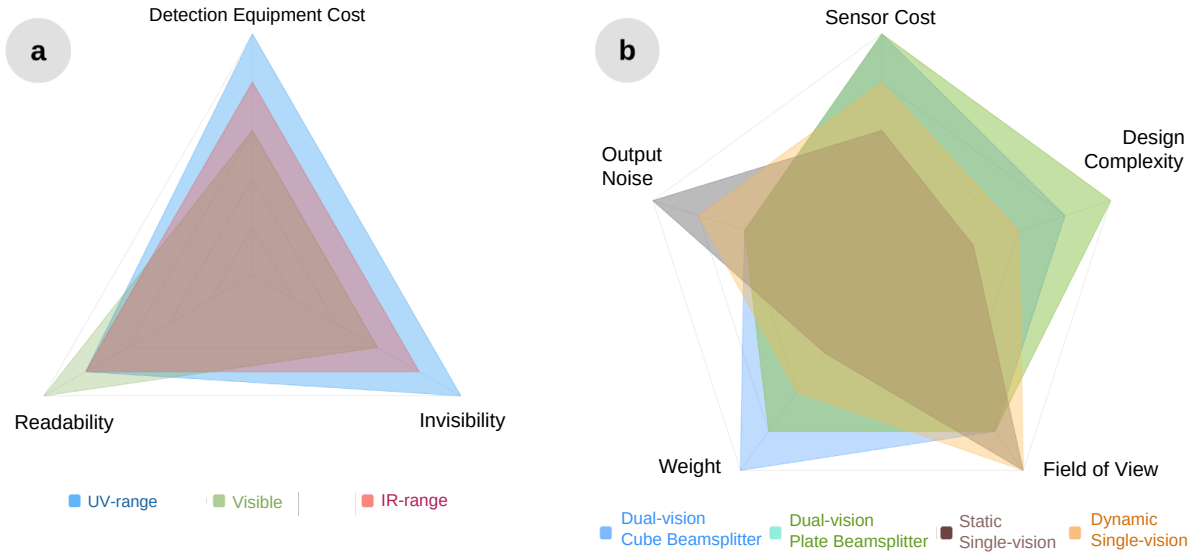


Figure 4.9: Relative performance scores of iMarker variants (**a**) and sensors (**b**) across multiple aspects, with larger areas indicating higher values. The term "BS" stands for beamsplitter.

polarization. Dual-vision configurations are more complex, with the cube beamsplitter offering more reliable light division than the plate beamsplitter, which requires precise alignment for optimal splitting. Design complexity directly influences sensor cost. The static single-vision setup is the most economical, requiring only a polarizer, whereas the dynamic single-vision configuration incurs additional costs due to its polarization-control components. Dual-vision setups are generally more expensive: they require two cameras, a beamsplitter (either cube or plate), and a structural mounting cage to ensure precise alignment and stability.

Regarding the "field of view," single-vision setups provide a broader theoretical FoV (*e.g.,* 87° × 58° in the RealSense™ D435, shown in Fig. 4.6c). However, in dual-vision setups, the beamsplitter's physical dimensions and the supporting cage partially obstruct the cameras' FoV, reducing the effective viewing area. For instance, the configuration shown in Fig. 4.6c achieves an effective FoV of 57° × 54.5°. Another essential factor is *weight*; the static single-vision setup is the lightest, whereas the dynamic single-vision setup adds approximately 30 gr due to the function generator and dynamic polarizer. Thus, single-vision setups are more feasible for minimizing payload weight, *e.g.,* in drones, as dual-vision configurations are heavier. The combined weight of dual-vision setup components typically ranges from 150 to 350 gr, with the plate beamsplitter potentially reducing the overall sensor weight.

When choosing among sensor configurations, a key selection criterion is the trade-off between "detection robustness" and "clutter (noise) rejection," as each configuration's detection pipeline can be tailored to optimize performance for specific application scenarios. Dual-vision setups rely on spatial frame subtraction followed by thresholding, producing cleaner outputs by isolating CSR-coated iMarker regions while effectively suppressing non-marker backgrounds thanks to the differential visibility of CSRs between the two sensor views. However, they are more sensitive to artifacts caused by sensor noise, motion, illumination variations, and frame misalignment. In contrast, single-vision setups may retain more background clutter (noise pixels) alongside detected iMarkers. Accordingly, enhancing robustness and output quality is addressed differently for each

sensor variant. While strategies such as adaptive thresholding and active frame registration can better aid dual-vision setup algorithms, single-vision systems can benefit from lightweight morphological filtering to enhance detection quality without compromising system simplicity.

4.7.2 Potential Applications

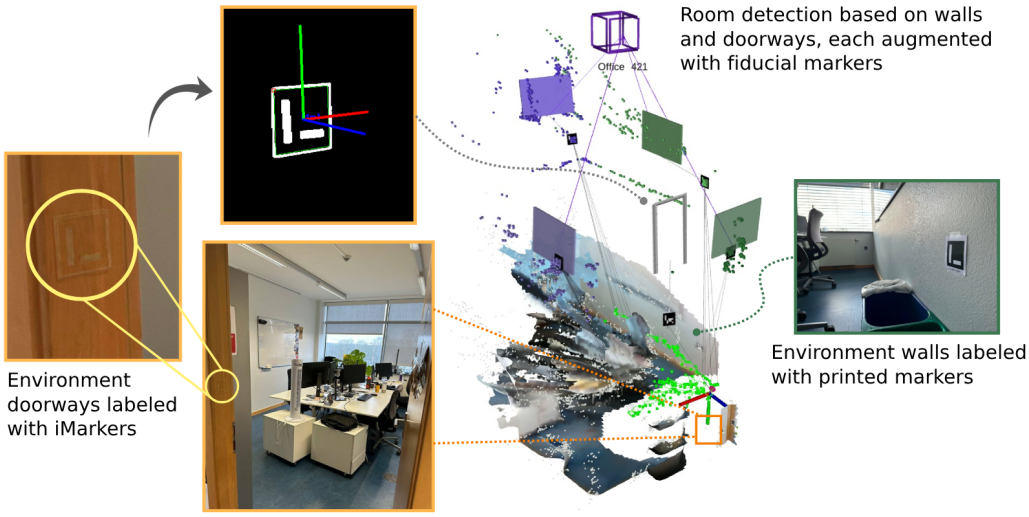
Considering the unique characteristics of iMarkers, they serve as potential alternatives in scenarios reliant on printed markers. In this regard, the obscure nature of iMarkers makes them appropriate for surroundings with unwanted visual clutter, such as museums, hospitals, offices, *etc.*, and for scenarios involving transparent or reflective structures (*e.g.,* glass walls and partitions), where placing printed fiducial markers is impractical or unacceptable due to their high visibility and disruption of visual transparency. By safely replacing printed markers, iMarkers can preserve unobtrusiveness and aesthetic integrity while delivering full functionality for robotics and AR systems. Note that various AR applications, particularly those deployed on handheld mobile devices, rely exclusively on RGB cameras and often use fiducial landmarks to support interactive features such as user engagement. This aligns well with the iMarker detection procedure proposed in this paper; specifically, the single-vision setup, which requires minimal hardware modifications to enable compatibility with standard AR-capable vision sensors.

Additionally, as the iMarker detection pipeline relies on simple computer vision techniques (*e.g.,* thresholding and frame subtraction), it introduces minimal computational overhead and operates efficiently even on a CPU-based system, without requiring hardware acceleration. Thus, the core innovation of the proposed system lies in tightly coupling hardware and software: the sensor hardware captures frames with filtered CSR-coated iMarker regions, and the software extracts and reveals these regions before passing the results to standard fiducial marker libraries (*e.g.,* ArUco or AprilTag).

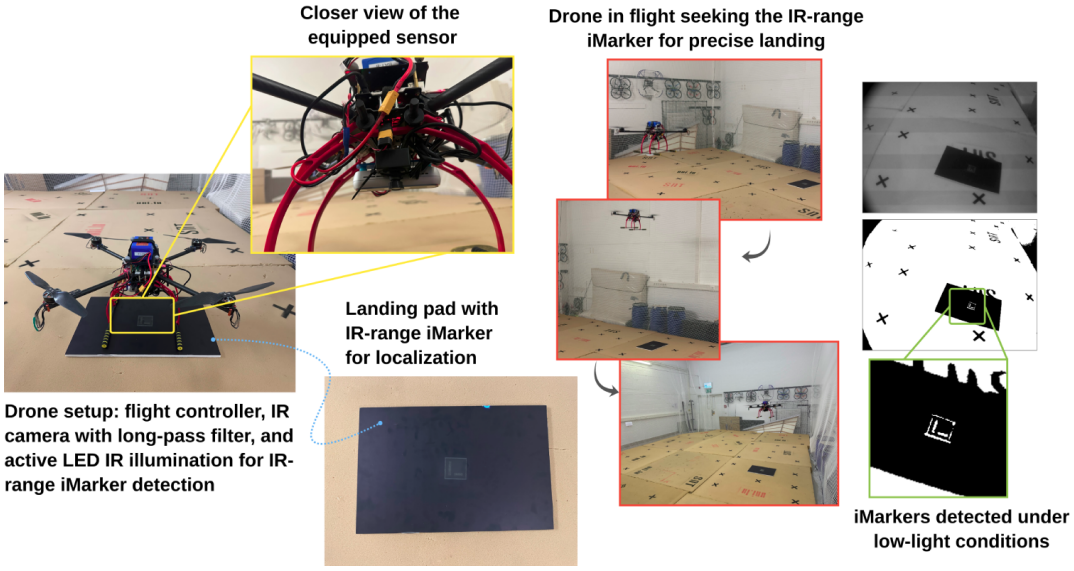
Fig. 4.10 depicts two representative real-world robotics applications in which iMarkers play a crucial role. In Fig. 4.10a, iMarkers are benchmarked within a VSLAM framework, showcasing the use of visible-range landmarks to support indoor map reconstruction enriched with semantic information [52]. Accordingly, iMarkers are applied to office door frames to unobtrusively encode semantic cues related to structural entities, thereby enabling semantic mapping while remaining visually camouflaged within the environment. In another setting, Fig. 4.10b shows a UAV-based robot-landing scenario introduced in Section 4.6.4, where conventional printed markers become unreliable or undetectable under low-illumination conditions, while IR-range iMarkers remain robustly detectable, enabling reliable localization and safe landing despite severely degraded visible-light cues.

4.7.3 Limitations and Fabrication Considerations

Given the ongoing nature of this research and the absence of a rapid iMarker fabrication process to date, the experiments remain constrained by varying illumination conditions, weather (for outdoor applications), and environmental factors (*e.g.,* dirt). In terms of environmental robustness, iMarkers are expected to be susceptible to dirt and occlusion, as with conventional printed markers, since both rely on visual input for detection and pose estimation. In other words, the marker recognition pipeline (ArUco in this work) processes either RGB frames from standard cameras or the binary outputs of the iMarker detector, without introducing additional sensitivity. Regarding sunlight and outdoor use, UV-range iMarkers may experience reduced contrast under ambient sunlight, potentially leading to false positives due to uncontrolled UV reflections. Accordingly, the reflectivity



(a) iMarkers integrated into a visual SLAM pipeline to support camera localization and semantic annotation of indoor environments [52].



(b) iMarkers deployed as low-illumination landing targets, enabling reliable pose estimation and precise autonomous landing of aerial robots.

Figure 4.10: Real-world robotics applications enabled by iMarkers.

of UV-range CSRs needs to be enhanced to ensure that all UV-range iMarkers are reliably detected when illuminated by the light source. Similarly, IR-range iMarkers could be influenced by environmental IR sources. In real-world deployments, additional filtering strategies or controlled illumination may be required to ensure reliable iMarker detection.

The scalability of iMarker fabrication for mass deployment remains an active area of investigation. Currently, production involves several manual steps, including microfluidic droplet generation, annealing, polymerization, and sequential washing, thereby reducing throughput and introducing variability in marker quality. To overcome this, we are developing an automated production platform that combines all stages into a continuous process with real-time computer monitoring. This system is expected to streamline manufacturing, improve consistency, and enable large-scale high-throughput fabrication suitable for real-world robotics applications.

4.8 Conclusions

This paper introduced a novel generation of fiducial markers, termed *iMarkers*, designed explicitly for real-world robotics applications. iMarkers are designed to be visually indistinguishable to the human eye, ensuring minimal environmental disruption while remaining fully detectable and recognizable by robots equipped with appropriate sensors and detection algorithms. The paper also presents various sensor designs and implemented algorithms for detecting iMarkers placed in the environment and exploiting them for different use cases. Experimental results demonstrate the practicality and potential of these markers as versatile landmarks for data embedding in various real-world scenarios.

Future work includes optimizing the iMarker fabrication process to ensure scalable production and enhancing illumination equipment, sensor designs, and algorithm implementations to ensure robustness across various scenarios.

5

Visual SLAM Survey

The second paper [166] was accepted for publication in **MDPI Sensors** in November 2022. The paper discusses current state-of-the-art approaches, their focus and trends, as well as potential directions for Visual SLAM, categorizing recent works and employed datasets into novel classes. The work has been selected as **Editor's Choice**.

Paper Card

Title: Visual SLAM: What Are the Current Trends and What to Expect?
Authors: A. Tourani, H. Bavle, J.L. Sanchez-Lopez, and H. Voos
Venue: MDPI Sensors
DOI: https://doi.org/10.3390/s22239297

Copyright

© 2026 by the authors. This article is an open-access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/.)

5.1 Abstract

In recent years, Simultaneous Localization and Mapping (SLAM) systems have shown significant performance, accuracy, and efficiency gains. In this regard, Visual SLAM (VSLAM) methods refer to the SLAM approaches that employ cameras for pose estimation and map reconstruction and are preferred over Light Detection And Ranging (LiDAR)-based methods due to their lighter weight, lower acquisition costs, and richer environment representation. Hence, several VSLAM approaches have evolved using different camera types (*e.g.,* monocular or stereo), and have been tested on various datasets (*e.g.,* TUM RGB-D or EuRoC) and in different conditions (*i.e.,* indoors and outdoors), and employ multiple methodologies to have a better understanding of their surroundings. The mentioned variations have made this topic popular for researchers and have resulted in various methods. In this regard, the primary intent of this paper is to assimilate the wide range of works in VSLAM and present their recent advances, along with discussing the existing challenges and trends. This survey is worthwhile to give a big picture of the current focuses in robotics and VSLAM fields based on the concentrated resolutions and objectives of the state-of-the-art. This paper provides an in-depth literature survey of fifty impactful articles published in the VSLAM domain. The mentioned manuscripts have been classified by different characteristics, including the novelty domain, objectives, employed algorithms, and semantic level. The paper also discusses the current trends and contemporary directions of VSLAM techniques that may help researchers investigate them.

5.2 Introduction

Simultaneous Localization and Mapping (SLAM) refers to the process of estimating an unknown environment's map while monitoring the location of an *agent* at the same time [175]. Here, the *agent* can be a domestic robot [41], an autonomous vehicle [42], a planetary rover [43], even an Unmanned Aerial Vehicle (UAV) [44, 176] or an Unmanned Ground Vehicle (UGV) [45]. In situations where a prior map of the environment is unavailable, or the robot's location is unknown, SLAM can be utilized to cover a wide range of applications. In this regard, and considering the ever-growing applications of robotics, SLAM has gained massive attention among industry and research community members in recent years [40, 39].

SLAM systems may use various sensors to collect data from the environment, including Light Detection And Ranging (LiDAR)-based, acoustic, and vision sensors [177]. The vision sensors category covers any variety of visual data detectors, including monocular, stereo, event-based, omnidirectional, and Red Green Blue-Depth (RGB-D) cameras. A robot equipped with a vision sensor uses the visual data provided by cameras to estimate the position and orientation of the robot with respect to its surroundings [178]. The process of using vision sensors to perform SLAM is particularly called Visual SLAM (VSLAM). Utilizing visual data in SLAM applications has the advantages of cheaper hardware requirements, more straightforward object detection and tracking, and the ability to provide rich visual and semantic information [47]. The captured images (or video frames) can also be used for vision-based applications, including semantic segmentation and object detection, as they store a wealth of data for processing. The mentioned characteristics have recently made VSLAM a trending topic in robotics and prompted robotics and Computer Vision (CV) experts to perform considerable studies and investigations in the last decades. Consequently, VSLAM can be found in various

types of applications where it is essential to reconstruct the 3D model of the environment, such as autonomous, AR, and service robots [179].

As a general benchmark introduced by [99] to tackle high computational cost, SLAM approaches mainly contain two introductory threads to be executed in parallel, known as *tracking* and *mapping*. Hereby, a fundamental classification of the algorithms used in VSLAM is how researchers employ distinct methods and strategies in each thread. The mentioned solutions look differently at SLAM systems based on the type of data they use, making them dividable into two categories: *direct* and *indirect (feature-based)* methods [96]. *Indirect* methods extract feature points (*i.e.,* key points) obtained from textures by processing the scene and keeping track of them by matching their descriptors in sequential frames. Despite the computationally expensive performance of feature extraction and matching stages, these methods are precise and robust against photometric changes in frame intensities. *Direct* algorithms, on the other hand, estimate camera motions directly from pixel-level data and build an optimization problem to minimize the photo-metric error. By relying on photogrammetry, these methods utilize all camera output pixels and track their replacement in sequential frames regarding their constrained aspects, such as brightness and color. These characteristics enable *direct* approaches to model more information from images than *indirect* techniques and enable a higher-accuracy 3D reconstruction. However, while direct methods work better in texture-less environments and do not require more computation for feature extraction, they often face large-scale optimization problems [97], and various lighting conditions negatively impact their accuracy. The pros and cons of each approach encouraged researchers to think about developing *hybrid* solutions, where a combination of both approaches is considered. *Hybrid* methods commonly integrate the detection stage of *indirect* and *direct*, in which one initializes and corrects the other.

Additionally, as VSLAMs mainly include a Visual Odometry (VO) front-end to locally estimate the path of the camera and a SLAM back-end to optimize the created map, the variety of modules used in each category results in implementation variations. VO provides a preliminary estimation of the location and poses of the robot based on local consistencies, which are sent to the back-end for optimization. Thus, the primary distinction between VSLAM and VO is whether or not to take into account the global consistency of the map and the predicted trajectory. Several state-of-the-art VSLAM applications also include two additional modules: *loop closure detection* and *mapping* [96]. They are responsible for recognizing previously visited locations for more precise tracking and map reconstruction based on the camera pose.

To summarize, Fig. 5.1 shows the overall architecture of a standard VSLAM approach. Accordingly, the system's inputs may also integrate with other sensor data, such as the IMU and LiDAR, to provide more information rather than visual data. Moreover, regarding the *direct* or *indirect* methodology used in a VSLAM pipeline, the functionality of the visual feature processing module might be changed or ignored. For instance, the "Feature Processing" stage is only employed in *indirect* approaches to process visual features obtained from the scene. Another factor is utilizing particular modules, such as loop closing detection and bundle adjustment, for improved execution.

This paper surveys **fifty VSLAM works** and classifies them into various categories according to diverse aspects. The authors hope their work will present a reference for the robotics community researchers working to improve VSLAM techniques. The rest of the paper is organized as follows: Other published surveys are introduced and discussed in the VSLAM domain in Section 5.3. Section 5.4 reviews the evolutionary stages of

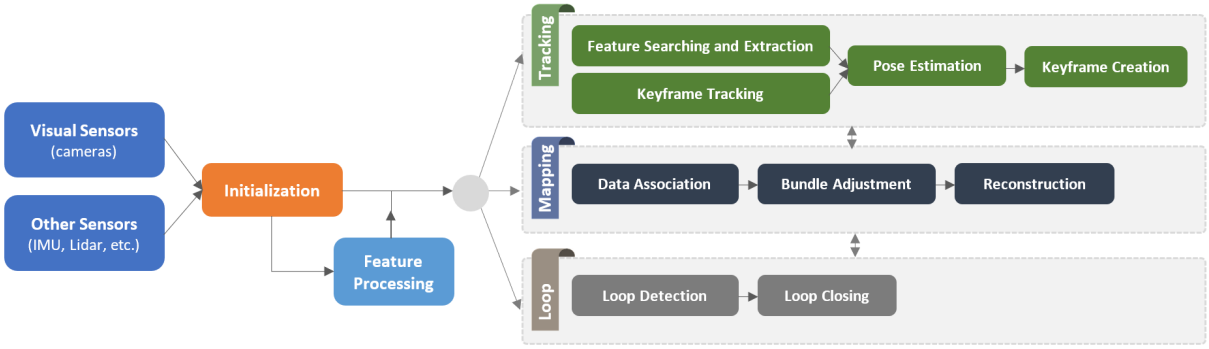


Figure 5.1: The flowchart of a standard visual SLAM approach. Regarding the direct/indirect methodology utilized, the functionality of some of these modules may change or be ignored.

VSLAM methods that led to the currently existing systems. An abstract level of various VSLAM modules is presented in Section 5.5, and a classification of the state-of-the-art based on the main contributions is available in Section 5.6. The authors will then discuss unresolved challenges and potential trends in this field in Section 5.7. The paper finally concludes in Section 5.8.

5.3 Related Surveys

There are various survey papers available in the domain of VSLAM that present a general review of the different existing approaches. Each of these papers reviews the major advantages and disadvantages of employing VSLAM approaches. Macario Barros *et al.* [180] divided approaches into three different classes: visual-only (monocular), visual-inertial (stereo), and RGB-D. They also proposed various criteria for simplifying and analyzing VSLAM algorithms. However, they did not include other vision sensors, such as event camera-based ones, which will be discussed later in Section 5.5.1. Chen *et al.* [46] reviewed a wide range of traditional and semantic VSLAM publications. They divided the SLAM development era into *classical*, *algorithmic-analysis*, and *robust-perception* stages, introducing key issues at each stage. They also summarized classical frameworks that employ direct/indirect methodologies and investigated the impact of deep learning algorithms on semantic segmentation. Although their work provides a comprehensive study of the advanced solutions in this domain, the classification of approaches is only restricted to the *feature types* employed in feature-based VSLAMs. Jia *et al.* [181] surveyed numerous manuscripts and presented a brief comparison between graph optimization-based methods and deep learning-equipped approaches. Despite presenting a proper comparison among various works, their discussion cannot be generalized due to the review of a limited number of papers. In another work, Abaspur Kazerouni *et al.* [182] covered various VSLAM methods, utilized sensory equipment, datasets, and modules, and simulated several indirect approaches for comparison and analysis. However, they contribute only to the feature-based algorithms—*e.g., Histogram of Oriented Gradients (HOG), Scale-Invariant Feature Transform (SIFT), and Speeded Up Robust Features (SURF)*—and deep learning-based solutions, and do not cover direct techniques. Bavle *et al.* [163] analyzed the situational awareness aspects in various SLAM and VSLAM applications and discussed their missing points. They could conclude that operating the lacking situational awareness features could enhance the performance of the current research works.

Other surveys that studied the latest VSLAM approaches focused only on a particular

topic or trend. For instance, Duan *et al.* [96] investigated the progress of deep learning in visual SLAM systems for transportation robotics. The authors summarized the advantages and drawbacks of utilizing various deep learning-based methods in VO and loop closure detection tasks in their paper. The significant advantage of using deep learning approaches in VSLAMs is the accurate feature extraction in pose estimation and the overall performance calculation. In another work in the same field, Arshad and Kim [183] focused on the impact of deep learning algorithms in loop closure detection using visual data. They reviewed various VSLAM papers and analyzed the long-term autonomy of robots in different conditions. Singandhupe and La [184] reviewed the impact of VO and VSLAM in driverless vehicles. They collected approaches that have been evaluated on the KITTI dataset [185], allowing for a brief overview of the advantages and disadvantages of each system. Cheng *et al.* [46] reviewed the VSLAM-based autonomous driving systems and raised the future development trends of such systems in a similar manuscript. Some other researchers have surveyed VSLAM works that can operate in real-world conditions. For instance, Saputra *et al.* [186] targeted the variations of VSLAM techniques operating in dynamic and rough environments and discussed the reconstruction, segmentation, tracking, and parallel execution of threads problems.

Regarding the mentioned surveys, the current survey has particularities that set it apart from other surveys presented so far and provides a comprehensive review of the VSLAM systems presented in different venues. With this, the **Research Questions (RQs)** that this survey aims to answer can be formulated according to the following dimensions:

- **RQ1:** How can the state-of-the-art methods be classified according to their part in the VSLAM domain and the objectives they try to achieve?
- **RQ2:** What contemporary directions of VSLAM methodologies have the recent publications focused on more?
- **RQ3:** What are the preferred evaluation criteria in recent works regarding the test environment and benchmarks?

The solutions to the first research question will be covered in Section 5.6, while **RQ2** and **RQ3** will be addressed in Section 5.7. In this regard, the major contributions of this survey compared to other available VSLAM surveys are:

- Categorizing various recent VSLAM publications regarding the main contributions, criteria, and objectives of researchers in proposing new solutions,
- Analyzing the current trends of VSLAM systems by profoundly investigating different approaches regarding dissimilar aspects,
- And introducing the potential contributions of VSLAM for researchers.

5.4 Evolution of Visual SLAM Algorithms

VSLAM systems have matured over the past years, and several frameworks have played a vital role in this development process. To provide a general picture, Fig. 5.2 illustrates the milestones of the widely referred to as VSLAM approaches that impacted the community and have been used as baselines for other frameworks.

Accordingly, the first endeavor in the literature to implement a real-time monocular VSLAM system was developed by Davison *et al.* in 2007, where they introduced a

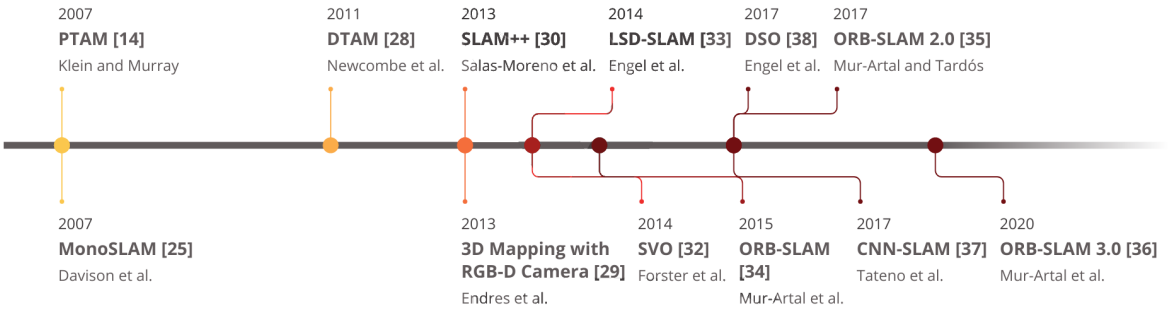


Figure 5.2: Milestones of highly impactful Visual SLAM approaches.

framework titled *Mono-SLAM* [98]. Their indirect framework could estimate the camera motion and 3D elements found in the world using the EKF algorithm [187]. *Mono-SLAM* began the primary action in the VSLAM domain despite the lack of global optimization and loop closure detection modules. However, the maps reconstructed by this method only include landmarks and do not offer further detailed information about the area. Klein *et al.* in [99] proposed Parallel Tracking and Mapping (PTAM) in the same year, in which they divided the entire VSLAM system into two primary threads: *tracking* and *mapping*. This multi-threading baseline was approved by many subsequent works, which will be discussed in this paper. The main idea of their approach was to reduce the computational cost and apply parallel processing to achieve real-time performance. While the *tracking* thread estimates camera motion in real-time, *mapping* predicts the 3D positions of feature points. PTAM was also the first approach to utilize BA for jointly optimizing the camera poses and the created 3D map. It uses the Features from Accelerated Segment Test (FAST) [188] corner detector algorithm for keypoint matching and tracking. Despite outperforming *Mono-SLAM*, the algorithm has a complex design and requires user input in the first stage.

A direct approach for measuring depth values and motion parameters for map construction was Dense Tracking and Mapping (DTAM) introduced by Newcombe *et al.* [100] in 2011. DTAM was a real-time framework equipped with *dense mapping* and *dense tracking* modules, and could determine camera poses by aligning the entire frames with a given depth map. To construct the environment map, the mentioned stages estimate the depth of the scene and the motion parameters, respectively. Although DTAM can provide a detailed presentation of the map, it has a high computational cost to perform in real time. As another indirect approach in the domain of 3D mapping and pixel-based optimization, Endres *et al.* [189], in 2013, proposed a method that could work with RGB-D cameras. Their method operates in real-time, focusing on low-cost embedded systems and small robots, but it struggles to produce accurate results in featureless or challenging scenarios. In the same year, Salas-Moreno *et al.* [190] proposed one of the first endeavors of utilizing semantic information in a real-time SLAM framework, titled *SLAM++*. Their system utilizes RGB-D sensor outputs to perform 3D camera pose estimation and tracking, thereby shaping a pose graph. A pose graph is a graph in which the nodes represent pose estimates and are connected by edges representing the relative poses between nodes with measurement uncertainty [191]. The predicted poses will then be optimized by merging the relative 3D poses obtained from the semantic objects in the scene.

5.5 VSLAM Setup

According to the previous section, which studied the evolution of VSLAM algorithms, the addition and improvement of various modules enhanced the architecture of existing techniques and enriched subsequent approaches. Thus, this section investigates multiple configurations that can be found in current VSLAM methodologies. Considering different setups, pipelines, and available configurations, current approaches can be classified into the categories mentioned below:

5.5.1 Sensors and Data Acquisition

The early-stage implementation of a VSLAM algorithm introduced by Davison *et al.* [98] was equipped with a monocular camera for trajectory recovery. **Monocular cameras** are the most common vision sensors for a wide range of tasks, such as object detection and tracking [192]. **Stereo cameras**, on the other hand, contain two or more image sensors, enabling them to perceive depth in the captured images, which leads to more accurate performance in VSLAM applications. These camera setups are cost-efficient and provide informative perception for higher accuracy demands. **RGB-D cameras** are other variations of visual sensors used in VSLAM and supply both the depth and colors in the scene. The mentioned vision sensors can provide rich information about the environment in straightforward circumstances—*e.g.,* proper lighting and motion speed—but they often struggle to cope with conditions where the illumination is low or the dynamic range in the scene is high.

In recent years, **event cameras** have also been used in various VSLAM applications. These low-latency bio-inspired vision sensors generate pixel-level brightness changes instead of standard intensity frames when a motion is detected, leading to a high dynamic range output with no motion blur impact [193]. In contrast to standard cameras, event-based sensors provide reliable visual information during high-speed motions and wide-range dynamic scenarios, but they fail to provide sufficient information when the motion rate is low. Although event cameras can outperform standard visual sensors in severe illumination and dynamic range conditions, they mainly generate unsynchronized information about the environment. This makes traditional vision algorithms unable to process the outputs of these sensors [194]. Additionally, using the spatio-temporal windows of events along with the data obtained from other sensors can provide rich pose estimation and tracking information.

Moreover, some approaches use a **multi-camera setup** to counter the common issues of working in a real-world environment and improve localization precision. Utilizing multiple visual sensors aids in situations where complicated problems, such as occlusion, camouflage, sensor failure, or sparsity of trackable texture, occur by providing cameras with overlapping fields of view. Although multi-camera setups can resolve some data acquisition issues, camera-only VSLAM may face various challenges, such as motion blur when encountering fast-moving objects, feature mismatching in low or severe illumination, dynamic object ignorance in scenarios with high pace changes, *etc.* Hence, some VSLAM applications may be equipped with multiple sensors alongside cameras. Fusing the events and standard frames [105] or integrating other sensors, such as LiDARs [195] and IMUs, to VSLAM are some of the existing solutions.

5.5.2 Visual Features

As discussed in Section 5.2, detecting visual features and utilizing feature descriptor information for pose estimation is an inevitable stage of indirect VSLAM methodologies.

These approaches utilize various feature extraction algorithms to better understand the environment and track feature points across consecutive frames. The feature extraction stage contains a wide range of algorithms, including SIFT [196], SURF [197], FAST [188], Binary Robust Independent Elementary Features (BRIEF) [198], ORB [199], *etc.* Among them, ORB features have the advantage of fast extraction and matching without losing huge accuracy compared to SIFT and SURF [200].

The problem with some of the mentioned methods is that they cannot effectively adapt to various complex and unforeseen situations. Thus, many researchers have employed Convolutional Neural Networks (CNNs) to extract deep-seated features of images for various stages, including VO, pose estimation, and loop closure detection. These techniques may represent supervised or unsupervised frameworks according to the functionality of the methods.

5.5.3 Target Environments

As a strong presumption in many traditional VSLAM practices, the robot works in a static world with no sudden or unanticipated changes. Consequently, although many systems could demonstrate a successful application in specific settings, some unexpected changes in the environment (*e.g.,* the existence of moving objects) are likely to cause complications for the system and degrade the state estimation quality to a large extent. Systems that work in dynamic environments usually employ algorithms such as Optical Flow or RANSAC [79] to detect movements in the scene, classify the moving objects as outliers, and skip them while reconstructing the map. Such systems utilize either *geometry/semantic* information or try to improve the localization scheme by combining the results of these two [128].

Additionally, different environments are classified into *indoor* and *outdoor* categories as a general taxonomy. An *outdoor* environment can be an *urban* area with structural landmarks and massive motion changes, such as buildings and road textures, or an *off-road* zone with a weak motion state, such as moving clouds and vegetation, the texture of the sand, *etc.* As a result of this, the amount of trackable points in *off-road* environments is less than in the *urban* areas, which increases the risk of localization and loop closure detection failure. *Indoor* environments, on the other hand, contain scenes with entirely different global spatial properties, such as corridors, walls, and rooms. It should be anticipated that while a VSLAM system might work well in one of the mentioned zones, it might not show the same performance in other environments.

5.5.4 System Evaluation

While some VSLAM approaches, particularly those capable of operating in dynamic and challenging environments, are tested on robots in real-world conditions, many research works have utilized publicly available datasets to demonstrate their applicability. In this regard, the **RAWSEEDS Dataset** by Bonarini *et al.* [201] is a well-known multi-sensor benchmarking tool containing indoor, outdoor, and mixed robot trajectories with ground-truth data. It is one of the oldest publicly available benchmarking tools for robotic and SLAM purposes. **Scenenet RGB-D** by McCormac *et al.* [202] is another favored dataset for scene understanding problems, such as semantic segmentation and object detection, containing five million large-scale rendered RGB-D images. The dataset also contains pixel-perfect ground-truth labels and exact camera poses and depth data, making it a potent tool for VSLAM applications. Many recent works in the domain of VSLAM and VO have tested their approaches on the **Technische Universität München (TUM)**

RGB-D dataset [167]. The mentioned dataset and benchmarking tool contain color and depth images captured by a *Microsoft Kinect* sensor and their corresponding ground-truth sensor trajectories. Further, **NTU VIRAL** by Nguyen *et al.* [203] is a dataset collected by a UAV equipped with a 3D LiDAR, cameras, IMUs, and multiple Ultra-widebands (UWBs). The dataset contains indoor and outdoor instances and is targeted for evaluating autonomous driving and aerial operation performances.

Moreover, the **European Robotics Challenge (EuRoC) Micro Aerial Vehicle (MAV)** dataset by Burri *et al.* [204] is another popular dataset containing images captured by a stereo camera, along with synchronized IMU measurements and motion ground truth. The collected data in *EuRoC MAV* are classified into easy, medium, and difficult categories according to the surrounding conditions. **OpenLORIS-Scene** by Shi *et al.* [205] is another publicly available dataset for VSLAM works, containing a wide range of data collected by a wheeled robot equipped with various sensors. It provides proper data for monocular and RGB-D algorithms, along with odometry data from wheel encoders. As a more general-purpose dataset used in VSLAM applications, **KITTI** [185] is a popular collection of data captured by two high-resolution RGB and grayscale video cameras on a moving vehicle. *KITTI* provides accurate ground truth using Global Positioning System (GPS) and laser sensors, making it a highly popular dataset for evaluation in mobile robotics and autonomous driving domains. TartanAir [206] is another benchmarking dataset for the evaluation of SLAM algorithms under challenging scenarios. Additionally, the **Imperial College London and National University of Ireland Maynooth (ICL-NUIM)** [207] dataset is another VO dataset containing handheld RGB-D camera sequences, considered a benchmark for many SLAM works.

In contrast with the previous datasets, some other datasets contain data acquired using particular cameras instead of regular ones. For instance, the **Event Camera Dataset** introduced by Mueggler *et al.* [208] is a dataset with samples collected using an event-based camera for high-speed robotic evaluations. Dataset instances contain inertial measurements and intensity images captured by a motion-capture system, making it a suitable benchmark for VSLAMs equipped with event cameras.

The mentioned datasets are used in multiple VSLAM methodologies according to their sensor setups, applications, and target environments. These datasets primarily consist of cameras' extrinsic and intrinsic calibration parameters, along with ground-truth data. The summarized characteristics of the datasets and some instances of each are shown in Table 5.1 and Fig. 5.3, respectively.

5.5.5 Semantic Level

Semantic information is required for the robot to understand the scene around it and make more profitable decisions. In many recent VSLAM works, adding semantic-level information to the geometry-based data is preferred to the pure geometry-based approaches, enabling them to deliver conceptual knowledge of the surroundings [127]. In this regard, a pre-trained object recognition module can add semantic information to the VSLAM models [209]. One of the most recent approaches is employing CNNs in VSLAM applications. In general, semantic VSLAM approaches contain four primary components described below [195]:

- **Tracking module:** It uses the two-dimensional feature points extracted from consecutive video frames to estimate the camera pose and construct three-dimensional map points. The calculation of the camera pose and construction of the 3D map points build the baselines of the localization and mapping processes, respectively.



Figure 5.3: Instances of some of the most popular visual SLAM datasets used for evaluation in various papers. The characteristics of these datasets can be found in Table 5.1.

- **Local mapping module**: By processing two sequential video frames, a new $3D$ map point is created, which is used along with a BA module for an improved camera pose.
- **Loop closing module**: By comparing keyframes with extracted visual features and assessing their similarities, the module tunes the camera pose and optimizes

the constructed map.

- **Non-Rigid Context Culling (NRCC)**: the primary goal of employing NRCC is to filter temporal objects from video frames to reduce their detrimental impact on the localization and mapping stages. It primarily features a masking/segmentation process for distinguishing various unstable instances in frames, such as people. Since it leads to a lower number of feature points to be processed, NRCC simplifies the computational part, resulting in more robust performance.

Accordingly, utilizing the semantic level in VSLAM approaches can improve the uncertainty of pose estimation and map reconstruction. However, the current challenge is to correctly utilize the extracted semantic information without significantly impacting the computational cost.

5.6 VSLAM State-of-the-Arts

Various researchers have proposed their approaches and contributions to the VSLAM community in recent years regarding the presented setups studied in the previous section. This section investigates how state-of-the-art techniques can be classified according to their contributions to the VSLAM domain (**RQ1**). To pinpoint VSLAM approaches that achieve rich outcomes and present robust architectures, the authors collected and filtered out highly cited publications published in top-notch venues in recent years from Google Scholar (`https://scholar.google.com/`, accessed on 15 October 2022) and well-known Computer Science bibliography databases: Scopus (`https://www.dblp.org/`, accessed on 15 October 2022) and DBLP (`https://www.scopus.com`, accessed on 15 October 2022). The manuscripts mentioned in the publications studied, particularly those most relevant to the VSLAM domain, have been purified. After exploring the papers, the collected publications are categorized based on their primary objectives to solve particular problems into sub-sections presented below:

5.6.1 Objective I: Multi-Sensor Processing

This category covers the range of VSLAM approaches that employ several sensors instead of a single camera to acquire data and understand the environment better. In this regard, while some techniques rely on *only cameras* as the operated visual sensors, others combine *various sensors* to enhance the accuracy of their algorithms.

Table 5.1: Commonly used datasets for VSLAM applications. *GT* in the table refers to the availability of ground-truth values.

Dataset Name	**Year**	**Environment**		**Utilized Sensors**									**GT**
		Indoor	Outdoor	GPS	LiDAR	IMU	Mono	Stereo	RGB-D	Event	Omni	UWB	
RAWSEEDS [201]	2006	✓	✓	✓	✓	✓		✓			✓		✓
KITTI [185]	2012		✓	✓	✓	✓	✓	✓					✓
ICL-NUIM [207]	2014	✓							✓				✓
TUM RGB-D [167]	2016	✓				✓			✓				✓
EuRoC MAV [204]	2016	✓				✓	✓	✓					✓
Event Camera [208]	2017		✓			✓				✓			✓
SceneNet RGB-D [202]	2017	✓							✓				✓
OpenLORIS [205]	2020	✓			✓	✓	✓	✓	✓				✓
TartanAir [206]	2020	✓	✓	✓			✓	✓	✓				✓
NTU VIRAL [203]	2021	✓	✓	✓		✓	✓					✓	✓

I-A. Multiple Cameras

Since it may be challenging to accurately capture the 3D trajectories of moving objects with a single camera, some researchers propose using multiple cameras instead. For instance, *CoSLAM* (`https://github.com/danping/CoSLAM`, accessed on 15 October 2022) is a VSLAM system introduced by Zou and Tan [101] that uses separate cameras deployed on various platforms to reconstruct robust maps. Their system combines multiple cameras moving around independently in a dynamic environment and reconstructs the map regarding their overlapping fields of view. The process makes it easier to rebuild dynamic points in $3D$ by mixing intra- and inter-camera pose estimation and mapping. *CoSLAM* tracks visual features using the Kanade-Lucas-Tomasi (KLT) algorithm and operates in static and dynamic contexts, including indoors and outdoors, where the relative positions and orientations may shift over time. The primary drawback of this method is that it requires sophisticated hardware to interpret numerous camera outputs, thereby increasing computational cost with each additional camera.

For challenging off-road settings, Yang *et al.* [210] developed a multi-camera cooperative panoramic vision VSLAM approach. Their approach gives each camera independence to increase the performance of the VSLAM system under challenging conditions, such as occlusion and texture sparsity. In order to determine the matching range, they extract ORB features from the cameras' overlapping fields of view. Additionally, they employed a deep learning technique based on a CNN to recognize similar features for loop closure detection. For the experiment, the authors used a dataset produced by a panoramic camera and an integrated navigation system.

MultiCol-SLAM is another open-source VSLAM framework with multi-camera configurations by Urban and Hinz [102]. They use their previously created model, *MultiCol*, to enhance ORB-SLAM, utilizing a keyframe-based process that supports multiple fisheye cameras. They added a Multi-Keyframes (MKFs) processing module to ORB-SLAM, which collects and turns images into keyframes. The authors also proposed the idea of multi-camera loop closing, in which loop closures are detected from MKFs. Although their method operates in real-time, it requires significant computer power because several threads must run simultaneously.

I-B. Multi-Modal Sensors

Other approaches proposed fusing various sensor modalities and combining vision- and inertial-based sensor outputs for better performance. In this regard, a low-cost, indirect LiDAR-assisted VSLAM called *CamVox* (`https://github.com/ISEE-Technology/CamVox`, accessed on 15 October 2022) was proposed by Zhu *et al.* [103] and demonstrated reliable performance and accuracy. Their method uses ORB-SLAM 2.0 and combines the unique capabilities offered by Livox LiDARs as the premium depth sensors with the outputs from RGB-D cameras. The authors used an IMU to synchronize and correct the non-repeating scanned locations. Their contribution is presenting an autonomous LiDAR-camera calibration method that operates in uncontrolled environments. Real-world tests on a robot platform indicate that *CamVox* performs in real time while processing the environment.

The authors of [104] proposed a multi-modal system titled *VIRAL (Visual-Inertial-Ranging-LiDAR) SLAM* that couples camera, LiDAR, IMU, and UWB. They also presented a map-matching marginalization scheme for visual features based on the local map constructed from LiDAR point clouds. The visual components are extracted and tracked using a BRIEF algorithm. The framework also contains a synchronization scheme and a

trigger for the utilized sensors. They tested their approach on simulation environments and their generated dataset, titled NTU VIRAL [203], which contains data captured by camera, LiDAR, IMU, and UWB sensors. However, their approach is computationally intensive due to handling synchronization, multi-threading, and sensor conflict resolution.

Vidal *et al.* [105] proposed integrating events, camera frames, and IMU in parallel configurations for reliable position estimation in high-speed settings. Their *Ultimate SLAM* (`https://github.com/uzh-rpg/rpg_ultimate_slam_open`, accessed on 15 October 2022) system is based on an event camera and a keyframe-based nonlinear optimization pipeline introduced in [211]. They use the FAST corner detector and the Lucas–Kanade tracking algorithm for feature detection and tracking, respectively. *Ultimate SLAM* avoids motion blur problems brought on by high-speed activity and operates in dynamic situations with varied lighting conditions. The efficiency of this technique on the "Event Camera Dataset" was evident in comparison to alternative event-only and conventional camera configurations. The authors also tested Ultimate SLAM on an autonomous quadrotor equipped with an event camera to show how their system can manage flight conditions that are impossible for conventional VO platforms to handle. The major challenge in *Ultimate SLAM* is the synchronization of events with standard frame outputs.

A tightly-coupled monocular camera and UWB range sensors were suggested by Nguyen *et al.* [212] for VSLAM. They use a combination of feature-based (visible) and featureless (UWB) landmarks to create a map. It operates effectively when UWB is exposed to multi-path effects in congested surroundings. They built their indirect method on ORB-SLAM and employed ORB features for pose estimation. They tested their system on a generated dataset with hand-carried movements simulating an employed aerial robot. The synchronization of the camera and UWB sensor is one of the difficulties in this case, but it has been overcome by employing a new camera pose with its related timestamp for each new image.

5.6.2 Objective II: Pose Estimation

Methods classified in this category focus on improving the pose estimation of VSLAM approaches using various algorithms. The primary patterns observed in this section utilize points and lines noticeable in the scene, employing particular features such as fiducial markers' data, and using deep learning to extract correct poses.

II-A. Lines/Points Data

In this regard, Zhou *et al.* [111] suggested employing building structural lines as practical features to determine the camera pose. Structural lines are associated with dominant directions and encode global orientation information, resulting in improved predicted trajectories. *StructSLAM*, the mentioned method, is a 6-DoF VSLAM technique that operates in both low-feature and featureless conditions. It employs EKF to estimate variables based on the current directions in the scene. For evaluation, the indoor scenes dataset from RAWSEEDS 2009 [201] and a set of generated sequential image datasets were used.

Point and Line SLAM (PL-SLAM) (`https://github.com/HarborC/PL-SLAM`, accessed on 15 October 2022), a VSLAM system based on ORB-SLAM optimized for non-dynamic low-texture settings, was introduced by Pumarola *et al.* [109]. The system simultaneously fuses line and point features for improved posture estimation and helps running in situations with few feature points. The authors tested *PL-SLAM* on their generated dataset and TUM RGB-D [167]. The drawback of their method is the com-

putational cost and the necessity of using other geometric primitives, *e.g.,* planes, for a more robust accuracy.

Gomez-Ojeda *et al.* [110] introduced *PL-SLAM* (`https://github.com/rubengooj/pl-slam`, accessed on 15 October 2022) (different from the framework with the same name by Pumarola *et al.* in [109]), an indirect VSLAM technique that uses points and lines in stereo vision cameras to reconstruct an unseen map. They merged segments obtained from points and lines in all VSLAM modules with visual information taken from successive frames in their approach. Using the ORB and Line Segment Detector (LSD) algorithms, points and line segments are retrieved and tracked in subsequent stereo frames in *PL-SLAM*. The authors tested PL-SLAM on EuRoC [204] and KITTI [185] datasets, outperforming the stereo version of ORB-SLAM 2.0 in terms of performance. One of the main drawbacks of *PL-SLAM* is the computational time required for the feature tracking module and considering all structural lines to extract information about the environment.

A degeneracy avoidance technique for monocular point- and line-based VSLAM systems was introduced by Lim *et al.* [112]. Another contribution of their work is introducing an Optical Flow-based line tracking module that extracts line characteristics, filters out the short lines in each frame, and matches the previously identified ones. To demonstrate the efficacy of their technique compared to the established point-based approaches, they tested their system on the EuRoC MAV [204] dataset. Their system lacks an adaptive approach to identify the correct optimization parameters, notwithstanding the strong findings.

II-B. Advanced Features

Dual Quaternion Visual SLAM (DQV-SLAM), a framework for stereo-vision cameras that uses a broad Bayesian framework for 6-DoF posture estimation, was proposed in [113]. To prevent the linearization of the nonlinear spatial transformation group, their approach employs progressive Bayes updates. For point clouds of maps and Optical Flow, *DQV-SLAM* uses ORB features to enable reliable data association in dynamic circumstances. On the KITTI [185] and EuRoC [204] datasets, the method could estimate experiment results reliably. However, it lacks a probabilistic interpretation for the stochastic modeling of poses and is computationally demanding for sampling approximation-based filtering.

Muñoz-Salinas *et al.* [114] developed a technique using artificial squared planar markers to recreate a large-scale interior environment map. Their real-time *SPM-SLAM* system can solve the ambiguity issue of pose estimation using markers if at least two of them are visible in each video frame. They created a dataset with video sequences of markers placed in two rooms joined by a door for examination. Although *SPM-SLAM* is cost-effective, it only works when numerous planar markers are scattered around the area, while at least two are visible for marker connection recognition. Moreover, the ability of their framework to handle dynamic changes in the scene is not measured.

II-C. Deep Learning

In another approach, Bruno and Colombini [117] proposed *LIFT-SLAM*, which combines deep learning-based feature descriptors with conventional geometry-based systems. They expanded the ORB-SLAM system's pipeline and employed a CNN to extract features from images, using the learned features to provide more dense and precise matches. For purposes of detection, description, and orientation estimation, *LIFT-SLAM* fine-tunes a *Learned Invariant Feature Transform (LIFT)* deep neural network. Studies using the KITTI [185] and EuRoC MAV [204] datasets' indoor and outdoor instances revealed that *LIFT-SLAM* outperforms conventional feature-based and deep learning-based VSLAM

systems in terms of accuracy. However, the weaknesses of the method are its computationally intensive pipeline and unoptimized CNN design, which leads to near real-time performance.

Naveed *et al.* [118] proposed another deep learning-based VSLAM solution with a reliable and consistent module, even on routes with extreme turns. Their approach outperformed several VSLAMs using a deep reinforcement learning network trained on realistic simulators. Furthermore, they provided a baseline for active VSLAM evaluation and could properly generalize across actual indoor and outdoor environments. The network's path planner developed the ideal path data, which is received by its base system, ORB-SLAM. The authors also released a dataset with actual navigation episodes in challenging and texture-less environments for evaluation.

As another approach, *RWT-SLAM* is a deep feature matching-based VSLAM framework that the authors in [119] proposed for weakly textured situations. Their method, which is based on ORB-SLAM, is fed with feature masks from an enhanced LoFTR [213] algorithm for local image feature matching. A CNN architecture and the LoFTR algorithm were used to extract coarse-level and fine-level descriptors in the scene, respectively. *RWT-SLAM* is examined on the TUM RGB-D [167] and OpenLORIS-Scene [205] datasets, as well as a real-world dataset gathered by the authors. However, their system is computationally demanding despite the robust feature-matching results and performance.

Sun *et al.* [120] proposed a VSLAM method that creates an object-oriented semantic point cloud map using integrated tracking and scene segmentation. Their deep learning-based system optimizes camera poses using object constraints and reconstructs multiple objects in 3D space. It is built upon ORB-SLAM 2.0 and utilizes Mask Region-based CNN (Mask R-CNN) [214] to extract 2D semantic information from the scene. Additionally, it uses the Deepsort [215] algorithm for multi-object tracking, which is detected by the semantic feature extractor. Experimental results on the TUM RGB-D dataset [167] after both self-training and using Microsoft COCO [216] weights showed deficient positioning errors. The authors compared their work with other semantic VSLAM methods, claiming their framework can extract semantic information more accurately than previous efforts. However, their system does not work in real-time due to demanding a pre-processing step for segmentation and the slow performance of Mask R-CNN.

5.6.3 Objective III: Real-World Viability

Approaches in this category have the primary objective of being used in various environments and working under several scenarios. The authors noticed that the references in this section are highly integrated with *semantic* information extracted from the environment and present an end-to-end VSLAM application.

III-A. Dynamic Environments

A VSLAM system titled *DS-SLAM* (`https://github.com/ivipsourcecode/DS-SLAM`, accessed on 15 October 2022), introduced by Yu *et al.* [127], can be used in dynamic contexts and offers semantic-level information for map construction. The system is built upon ORB-SLAM 2.0 and contains five threads: *tracking*, *semantic segmentation*, *local mapping*, *loop closing*, and *dense semantic map construction*. To exclude dynamic items before the pose estimation process and increase localization accuracy, *DS-SLAM* employs the Optical Flow algorithm with a real-time semantic segmentation network called *SegNet* [217]. *DS-SLAM* has been tested in real-world settings and with RGB-D cameras, as well as on the TUM RGB-D dataset [167]. However, despite its high localization accuracy, it faces semantic segmentation limitations and computationally intensive features.

Semantic Optical Flow SLAM (SOF-SLAM), an indirect VSLAM system built upon the RGB-D mode of ORB-SLAM 2.0, is another method in highly dynamic environments proposed by Cui and Ma [128]. Their approach uses the Semantic Optical Flow dynamic feature detection module, which extracts and skips the changing features concealed in the semantic and geometric information provided by ORB feature extraction. In order to deliver accurate camera pose and environment reports, *SOF-SLAM* makes use of SegNet's pixel-wise semantic segmentation module. In highly dynamic situations, experimental findings on the TUM RGB-D dataset and in real-world settings demonstrated that SOF-SLAM performs better than ORB-SLAM 2.0. However, the ineffective method of non-static feature recognition and reliance on just two consecutive frames for this purpose are *SOF-SLAM*'s weakest points.

Using the Optical Flow method to separate and eliminate dynamic feature points, Cheng *et al.* [134] suggested a VSLAM system for dynamic environments. They have utilized the ORB-SLAM pipeline's structure and supplied it with fixed feature points generated from typical monocular camera outputs for precise posture estimation. The system operates in featureless circumstances by sorting Optical Flow values and using them for feature recognition. According to experimental results on the TUM RGB-D dataset [167], the suggested system functions well in dynamic indoor circumstances. However, the system's configuration uses an offline threshold for motion analysis, making it difficult to use in a variety of dynamic environment situations.

Another VSLAM strategy was released by Yang *et al.* [135] that reconstructs the environment map using semantic segmentation network data, a motion consistency detection technique, and geometric restrictions. Their approach, which is based on ORB-SLAM 2.0's [107] RGB-D setting, performs well in dynamic and indoor environments. Only the stable features from the scene are retained using an improved ORB feature extraction technique, while the dynamic characteristics are disregarded. The features and the semantic data will then be combined to create a static semantic map. Evaluation findings on the Oxford and TUM RGB-D datasets demonstrated the effectiveness of their approach in enhancing location accuracy and creating semantic maps with a wealth of data. However, their system can run into problems in corridors or places with less information.

III-B. Deep Learning

In another work by Li *et al.* [121] called *DXSLAM* (`https://github.com/ivipsourcecode/dxslam`, accessed on 15 October 2022), deep learning is used to find keypoints that resemble SuperPoints and to produce both the general descriptors and the images' keypoints. They trained a cutting-edge deep CNN titled *HF-NET* to produce frame- and keypoint-based descriptions by extracting local and global information from each frame. They also used the offline Bag of Words (BoW) method to train a visual vocabulary of local characteristics for precise loop closure recognition. *DXSLAM* operates in real-time without using a Graphics Processing Unit (GPU) and is compatible with contemporary CPUs. Even if such qualities are not explicitly addressed, it has a remarkable ability to resist dynamic changes in dynamic contexts. *DXSLAM* has been tested on TUM RGB-D [167] and OpenLORIS-Scene [205] datasets and both indoor and outdoor images, and could achieve more accurate results than ORB-SLAM 2.0 and DS-SLAM. However, the major disadvantages of this method are the complex architecture for feature extraction and the incorporation of deep features into an old SLAM framework.

In another approach, Li *et al.* [122] developed a real-time VSLAM technique for extracting feature points based on deep learning in complicated situations. The method

can run on a GPU and supports the creation of 3D dense maps, and is a multi-task CNN for feature extraction with self-supervision capabilities. The CNN output is binary code strings with a fixed length of 256, making it possible to be replaced by more conventional feature point detectors such as ORB. It comprises three threads for reliable and timely performance in dynamic scenarios: *tracking*, *local mapping*, and *loop closing*. The system supports monocular and RGB-D cameras using ORB-SLAM 2.0 [107] as a baseline. The authors tested their methodology on the TUM dataset and two additional datasets, collected in a corridor and an office, using a Kinect camera for the experiments.

Steenbeek and Nex [123] present a real-time VSLAM technique that uses a ResNet-50 [218] CNN for accurate scene interpretation and map reconstruction. Their solution utilizes monocular camera streams from a UAV during flight and employs a depth-estimating neural network for reliable performance. The mentioned method is based on ORB-SLAM 2.0 and makes use of visual cues collected from indoor environments. Additionally, the CNN is trained on more than 48,000 indoor examples and operates on the pose, space depth, and RGB inputs to estimate scale and depth. The TUM RGB-D dataset and a real-world test using a drone were used to evaluate the system, which demonstrated enhanced pose estimation accuracy. However, the system struggles in situations without texture and needs both CPU and GPU resources for real-time performance.

Su *et al.* [124] proposed a fast deep learning-based VSLAM system for dynamic environments. Their real-time framework integrates ORB-SLAM 2.0 with the lightweight object detection network YOLOv5s [219], enabling it to acquire semantic information in the scene. Another contribution of their work is introducing an optimized homography matrix module to create more accurate Optical Flow vectors. According to the experiments, their method performs better than the existing semantic VSLAM algorithms in terms of accuracy and performance. They employed the TUM RGB-D dataset to analyze their system's runtime and accuracy compared to other works. Nevertheless, according to the authors' claims, the backbone network of YOLOv5 in their approach needs to be optimized yet to make it more robust and practical.

In another work, with a focus on resolving the challenges of dynamic scenes, Chen *et al.* [125] proposed a deep learning-based VSLAM approach. They presented a high-precision contour extractor and a lightweight contour optimization algorithm that improves instance segmentation accuracy at a low cost. Their method has been tested on the TUM RGB-D dataset, resulting in robust pose estimation and map construction compared to other VSLAM approaches, such as ORB-SLAM 2.0. Despite the mentioned advantages, whether the proposed approach works fine under complex circumstances is not verified.

III-C. Artificial Landmarks

A technique called *UcoSLAM* (`https://sourceforge.net/projects/ucoslam/`, accessed on 15 October 2022) [150] by Muñoz-Salinas and Medina-Carnicer outperforms conventional VSLAM systems by combining natural and human-made landmarks and automatically calculating the scale of the surroundings using fiducial markers. *UcoSLAM*'s primary driving force is to combat natural landmarks' instability, repetition, and poor tracking qualities. It can operate in surroundings without tags or features since it can operate in keypoints-only, markers-only, and mixed modes. *UcoSLAM*'s tracking mode locates map correspondences, optimizes re-projection errors, and re-localizes in the event of tracking failure. Additionally, it has a marker-based loop closure detection system and can describe features using any descriptor, including ORB and FAST. Despite all the plus points of *UcoSLAM*, the system executes in multiple threads, making it a time-consuming

approach for particular scenarios.

III-D. Multi-Modal Setups

Another VSLAM strategy for dynamic indoor and outdoor situations is *DMS-SLAM* [220], which supports monocular, stereo, and RGB-D visual sensors. The system employs sliding window and Grid-based Motion Statistics (GMS) [221] feature-matching methods to find static feature locations. Using the ORB-SLAM 2.0 system as its foundation, *DMS-SLAM* tracks the static features recognized by the ORB algorithm. The authors tested their suggested methodology on the TUM RGB-D [167] and KITTI [185] datasets and outperformed cutting-edge VSLAM algorithms. Additionally, because the feature points on the dynamic objects were removed during the tracking step, *DMS-SLAM* performs more quickly than the original ORB-SLAM 2.0. Despite the described positive points, the suggested solution encounters difficulties in situations with little texture, fast motion, and highly dynamic environments.

5.6.4 Objective IV: Resource Constraint

In another category, some of the VSLAM methodologies are built for devices with limited computational resources compared to other standard devices. Approaches in this category either attempt to deliver a lightweight setup enabling them to work in scenarios with constrained hardware or use complementary resources for data offloading. For instance, VSLAM systems designed for mobile devices and robots with embedded systems are included in this category.

IV-A. Limited Processing Capabilities

EdgeSLAM is a real-time, edge-assisted semantic VSLAM system for mobile and resource-constrained devices proposed by Xu *et al.* [143]. It employs a series of fine-grained modules to be used by an edge server and the associated mobile devices rather than requiring heavy threads. A semantic segmentation module based on *Mask R-CNN* is also included in *edgeSLAM* to improve segmentation and object tracking. The authors put their strategy into practice on an edge server with several commercial mobile devices, such as cell phones and development boards. By reusing the findings of the object segmentation, they avoided duplicate processing by adapting system parameters to different network bandwidth and latency situations. *EdgeSLAM* has been evaluated on monocular vision instances of TUM RGB-D, KITTI, and the created dataset for experimental settings.

For stereo camera setups, Schlegel, Colosi, and Grisetti [145] suggested a lightweight feature-based VSLAM framework titled *ProSLAM* (`https://gitlab.com/srrg-software/srrg_proslam`, accessed on 15 October 2022) that achieves results on par with cutting-edge techniques. Four modules make up their approach: the *triangulation* module, which creates 3D points and associated feature descriptors; the *incremental motion estimation* module, which processes two frames to determine the current position; the *map management* module, which creates local maps; and the *re-localization* module, which updates the world map based on the similarities of local maps. *ProSLAM* retrieves the 3D position of the points using a single thread and leverages a small number of well-known libraries for a system that is simple to create. According to the experiments on KITTI and EuRoC datasets, their approach can achieve robust results. However, it shows weakness in rotation estimation and does not contain any bundle adjustment modules.

Bavle *et al.* [146] proposed *VPS-SLAM* (`https://github.com/hridaybavle/semantic_slam`, accessed on 15 October 2022), a lightweight graph-based VSLAM framework for aerial robotics. Their real-time system integrates geometrical data, several object de-

tection techniques, and visual/visual-inertial odometry for pose estimation and building the semantic map of the environment. Low-level characteristics, IMU measurements, and high-level planar information are all used by VPS-SLAM to reconstruct sparse semantic maps and predict robot states. The system leverages the lightweight version of YOLO2 [222] trained on the COCO dataset [216] for object detection due to its real-time and computationally effective performance. They used a handheld camera setup and an aerial robotic platform equipped with an RGB-D camera for testing. The TUM RGB-D dataset's indoor instances were used to test their methodology, and they were able to provide results that were on par with those of well-known VSLAM methods. However, only a small number of objects (*e.g.,* chairs, books, and laptops) can be used by their VSLAM system to build a semantic map of the surrounding area.

Another real-time indoor VSLAM method was proposed by Tseng *et al.* [223] that requires a low-cost setup. The authors also presented a technique for estimating the number of frames and visual elements required for a reasonable degree of localization accuracy. Their solution is based on the OpenVSLAM [147] framework and makes use of it for emergencies that arise in the real world, such as gaining access to specific targets. The system acquires the scene's feature map for precise pose estimation by applying the Efficient Perspective-n-Point (EPnP) and RANSAC algorithms. According to tests conducted in a building, their device can deliver accurate findings under difficult lighting conditions.

IV-B. Computation Offloading

Ben Ali *et al.* [144] suggested using edge computing to enable the offloading of resource-intensive operations to the cloud and reduce the computational burden on the robot. They modified the architecture of ORB-SLAM 2.0 in their indirect framework, Edge-SLAM (`https://github.com/droneslab/edgeslam`, accessed on 15 October 2022), by maintaining the tracking module on the robot and delegating the remainder to the edge. By splitting the VSLAM pipeline between the robot and the edge device, the system can maintain both a local and a global map. With fewer resources available, they could still execute properly without sacrificing accuracy. They used the TUM RGB-D dataset and two different mobile devices to generate a custom indoor environment dataset using RGB-D cameras for evaluation. However, one of their approach's drawbacks is the architecture's complexity due to the decoupling of various SLAM modules. Another setback is that their system works only in short-term settings, and utilizing Edge-SLAM in long-term scenarios (*e.g.,* multiple days) would face performance degradation.

5.6.5 Objective V: Versatility

VSLAM works categorized in this class are focused on straightforward development, utilization, adaptation, and extension. In this regard, the practices encountered in such VSLAM frameworks are to simplify the development of functional changes for further improvements.

In this regard, Sumikura *et al.* [147] introduced *OpenVSLAM* (`https://github.com/xdspacelab/openvslam`, accessed on 15 October 2022), a highly adaptable open-source VSLAM framework that seeks to be quickly developed upon and called by other third-party programs. Their feature-based approach is compatible with multiple camera types, including monocular, stereo, and RGB-D, and can store or reuse the reconstructed maps for later usage. *OpenVSLAM* performs better in terms of tracking accuracy and efficiency than ORB-SLAM and ORB-SLAM 2.0 due to its powerful ORB feature extractor module. However, the open-source code of the system has been discontinued owing to worries over

code similarities that infringed on the rights to ORB-SLAM 2.0.

To bridge the gap between real-time capabilities, accuracy, and resilience, Ferrera *et al.* [148] developed *OV²SLAM* (`https://github.com/ov2slam/ov2slam`, accessed on 15 October 2022), which works with monocular and stereo-vision cameras. By limiting the extraction of features to keyframes and monitoring them in subsequent frames by eliminating photo-metric errors, their method lessens the computational load. In this sense, *OV²SLAM* is a hybrid strategy that combines the virtues of the direct and indirect categories of VSLAM algorithms. Using well-known benchmarking datasets, including EuRoC, KITTI, and TartanAir in both indoor and outdoor experiments, it was demonstrated that *OV²SLAM* surpasses several popular techniques in terms of performance and accuracy.

Another approach in this category, titled *DROID-SLAM* (`https://github.com/princeton-vl/DROID-SLAM`, accessed on 15 October 2022), a deep learning-based visual SLAM for monocular, stereo, and RGB-D cameras, is proposed by Teed and Deng [149]. They could attain greater accuracy and robustness than well-known monocular and stereo-track methods. Their solution operates in real-time and consists of *back-end* (for bundle adjustment) and *front-end* (for keyframe collection and graph optimization) threads. *DROID-SLAM* has already been taught using monocular camera examples. Therefore, it does not need to be trained again to use stereo and RGB-D inputs. The approach minimizes the projection error, like indirect methods, while not requiring any pre-processing for feature identification and matching. A feature extraction network comprising downsampling layers and residual blocks processes each input image to create dense features. *DROID-SLAM* has been tested on well-known datasets, including TartanAir, EuRoC, and TUM RGB-D, and could achieve acceptable results.

Bonetto *et al.* in [224] propose *iRotate* (`https://github.com/eliabntt/irotate_active_slam`, accessed on 15 October 2022), an active technique for omnidirectional robots with RGB-D cameras. Additionally, a module for spotting obstructions in the camera's area of vision is employed in their approach. By offering observation coverage of previously unexplored places and previously visited locations, *iRotate*'s primary objective is to lessen the distance the robot must go to map the environment. The mentioned method uses a VSLAM framework with graph features as its back-end. The authors could attain outcomes that were on par with those of cutting-edge VSLAM methods by providing comparisons in simulation and on a real three-wheel omnidirectional robot. However, the major weakness of their method is that the robot might face start-stop cases in which the local paths are re-planned.

5.6.6 Objective VI: Visual Odometry

Approaches in this category aim to determine the position and orientation of the robot with the highest possible accuracy. With this, the leading solutions are using deep neural networks to extract visual patterns, process adjacent frames' data for enhanced tracking, and utilize other feature processing methodologies.

VI-A. Deep Learning

The *Dynamic-SLAM* framework was proposed in [131], which leverages deep learning for accurate pose prediction and suitable environment comprehension. As part of a semantic-level module for optimized VO, the authors employed a CNN to identify moving objects in the environment, which helped them lower the pose estimate error brought on by improper feature matching. Additionally, *Dynamic-SLAM* uses a selective tracking module to ignore dynamic locations in the scene and a missed feature corrective algorithm for speed

invariance in adjacent frames. Despite the excellent results, the system requires huge computational costs and faces the risk of misclassifying dynamic/static objects due to a limited number of defined semantic classes.

Bloesch *et al.* [115] proposed the *Code-SLAM* (`https://github.com/silviutroscot/CodeSLAM`, accessed on 15 October 2022) direct technique, which offers a condensed and dense representation of the scene geometry. Their VSLAM system, which only functions with monocular cameras, is an enhanced version of *PTAM* [99]. They divided intensity images into convolutional features and fed them to a depth auto-encoder using a CNN trained on intensity images from the SceneNet RGB-D dataset [202]. Indoor examples of the EuRoC datasets have been used to test *Code-SLAM*, and the findings were promising in terms of accuracy and performance.

DeepVO (`http://senwang.gitlab.io/DeepVO/`, accessed on 15 October 2022), an end-to-end VO framework using a Deep Recurrent Convolutional Neural Network (RCNN) architecture for monocular settings, was proposed by Wang *et al.* [225]. Their approach uses deep learning to automatically learn the appropriate features, model sequential dynamics and relations, and infer poses directly from color frames. The *DeepVO* architecture includes a CNN called FlowNet for computing Optical Flow from sequential frames and two Long Short-Term Memory (LSTM) layers for estimating the temporal changes based on the feed provided by the CNN. The framework can simultaneously extract visual characteristics and perform sequential modeling by combining CNN and Recurrent Neural Network (RNN). *DeepVO* can incorporate geometry with the knowledge models learned for an enhanced VO. However, it cannot be utilized to replace conventional geometry-based VO approaches.

Parisotto *et al.* [226] proposed a DeepVO-like end-to-end system using a Neural Graph Optimization (NGO) step instead of LSTMs. Their approach operates a loop closure detection and correction mechanism based on temporally distinct poses. NGO uses two attention-optimization methods to jointly optimize the aggregated predictions made by the convolutional layers of local pose estimation modules and deliver global pose estimations. They experimented with their technique on $2D$ and $3D$ mazes and outperformed DeepVO's performance and accuracy levels. The mentioned approach needs to be connected to an SLAM framework to supply the re-localization signals.

In another work, one of the most extensive VSLAM frameworks titled *DeepFactors* (`https://github.com/jczarnowski/DeepFactors`, accessed on 15 October 2022) was introduced by Czarnowski *et al.* [227] for densely rebuilding the environment map with monocular cameras. For a more reliable map reconstruction, their real-time solution performs joint optimization of the pose and depth, makes use of probabilistic data, and combines learned and model-based approaches. The authors modified the *CodeSLAM* framework and added missing components, such as local/global loop detection. The system is evaluated on the ICL-NUIM and TUM RGB-D datasets after being trained on roughly 1.4 million ScanNet [202] images. *DeepFactors* improves the idea of the *CodeSLAM* framework and focuses on code optimization in traditional SLAM pipelines. However, due to the computational costs of the modules, this approach requires employing a GPU to guarantee real-time performance.

The authors of [126] proposed a VSLAM approach based on ORB-SLAM that operates well in low-texture and illumination-changing environments pertaining to its deep CNN algorithm. Their system utilizes a Visual Geometry Group (VGG) [228] network for accurate feature point extraction and an improved bundle adjustment optimization that operates pleasingly with VGG-based VO. A set of modifications was considered on the

standard VGG architecture to keep the approach real-time. They trained the network on an indoor image dataset collected by a wheeled robot equipped with a monocular camera and an IMU. The experiments showed that the proposed approach was very good when examining feature detection and matching functionalities in various illumination conditions. It should be noted that despite the accurate feature matching, the experiments needed to be verified on public datasets and outdoor settings.

Gu *et al.* [229] proposed a modified version of ORB-SLAM that uses deep binary local descriptors provided by a CNN instead of ORB features. Their approach, *DBLD-SLAM*, can improve positioning and VO accuracy by utilizing higher-quality feature matching and tracking, leading to enhanced adaptive scale, quantization, and correlation losses. Experiments on the Tartanair and HPatches [230] datasets showed that *DBLD-SLAM* achieves better results than other VSLAM methodologies with hand-crafted feature processing. Despite the real-time performance of the proposed system, some revisions are required to decrease its computational cost.

VI-B. In-Depth Adjacent Frame Processing

By reducing the photometric and geometric errors between two pictures for camera motion detection, the authors of [231] (`https://vision.in.tum.de/data/software/dvo`, accessed on 15 October 2022) developed a real-time dense SLAM approach for RGB-D cameras, improving their prior method [232]. Their keyframe-based solution expands *Pose SLAM* [233] that only keeps non-redundant poses for producing a compact map, adds dense visual odometry characteristics, and effectively utilizes the information from camera frames for a reliable camera motion estimation. The authors also employed an entropy-based technique to gauge keyframe similarity for loop closure detection and drift avoidance. However, their approach still needs work in the areas of loop closure detection and keyframe selection quality.

In another work introduced by Li *et al.* [234], real-time dynamic object removal is accomplished using a feature-based VSLAM approach known as *DP-SLAM*. This method uses a Bayesian probability propagation model based on the likelihood of the keypoints derived from moving objects. The variation of geometry restrictions and semantic data can be overcome by *DP-SLAM* using the moving probability propagation algorithm and iterative probability updates. It is integrated with ORB-SLAM 2.0 and has been tested on the TUM RGB-D dataset. Despite the accurate results, the system only works in sparse VSLAMs and faces high computational costs due to the iterative probability updater module.

Pair-Navi, a suggested indoor navigation system by Dong *et al.* [235], reuses a previously traced path by an agent for usage in the future by other agents. Hence, a previous traveler, known as the *leader*, captures the trace information, such as turnings and particular ambient qualities, and gives it to a later *follower* that needs to travel to the same destination. While the *follower* uses a re-localization module to determine its location concerning the reference trace, the *leader* incorporates visual odometry and trajectory creation modules. The system employs Mask R-CNN to recognize and remove dynamic objects from the video feature set. They tested *Pair-Navi* on a set of generated datasets and several smartphones for the experiments.

VI-C. Various Feature Processing

A text-based VSLAM system called *TextSLAM* proposed by Li *et al.* [116] incorporates text items retrieved from the scene using the FAST corner detection technique in the SLAM pipeline. Texts include a variety of textures, patterns, and semantic meanings,

making the approach more efficient to use them to create 3D text maps of high quality. *TextSLAM* uses texts as reliable visual fiducial markers, parametrizes them after the first frame in which they are found, and then projects the 3D text object onto the target image to locate it again. They also presented a new three-variable parameterization technique for initializing instantaneous text features. Using a monocular camera and a dataset created by the authors, experiments were conducted in indoor and outdoor settings. Operating in text-free surroundings, interpreting short letters, and requiring the storage of enormous text dictionaries are the three fundamental challenges of *TextSLAM*.

Xu *et al.* [195] proposed an indirect VSLAM system built upon a modified ORB-SLAM that enables high-accuracy localization and user interaction using the Occupancy Grid Mapping (OGM) method and a new 2D mapping module. Their system can reconstruct the environment map that shows the presence of a barrier as an equally spaced field of variables using the OGM, making it possible to navigate continuously and in real time while planning a route. The experimental examination of a generated dataset shows that their approach functions in GPS-denied conditions. However, their technique struggles to function well in dynamic and complicated environments and has trouble appropriately matching the features in corridors and featureless conditions.

CPA-SLAM, a direct VSLAM method for RGB-D cameras that uses planes for tracking and graph optimization, was proposed by Ma *et al.* [236]. *Frame-to-keyframe* and *frame-to-plane* alignments are regularly integrated into their technique. They also introduced an image alignment algorithm for tracking the camera with respect to a reference keyframe and aligning the image with the planes. The keyframe data are used by *CPA-SLAM*, which looks for the closest short temporal and geographical distances to track. The system's real-time performance of the tracking system was tested in with and without plane settings, with analyses performed on TUM RGB-D and ICL-NUIM datasets with indoor and outdoor scenes. However, it only supports a small number of geometric shapes, *i.e.,* planes.

5.7 Current Trends

This section discusses what the current trends are and what is better to invest in the VSLAM domain according to the broad range of research papers. These proposed systems cannot be recommended or rejected in this regard, as each is designed for a particular application and performs a specific objective. Thus, the authors concentrated on the shared aspects of the papers to extrapolate the hidden relationships between them and extract trendy topics. These aspects can include four main subjects: **1)** the presence of a base framework to develop a system upon it, **2)** the main target of the proposed solution, **3)** the importance of semantic information for better performance, and **4)** the evaluation criteria. The rest of this section introduces the statistics based on these four topics and analyzes the extracted outcomes.

5.7.1 Statistics

Regarding the classification of the surveyed papers in the various aspects presented above, the processed data are visualized in Fig. 5.4 to figure out the current trends in VSLAM. In Fig. 5.4a, the majority of the proposed VSLAM systems are standalone applications that implement the whole procedure of localization and mapping using visual sensors from scratch. While ORB-SLAM 2.0 [107] and ORB-SLAM [106] are other *base* platforms employed to make a new framework, minimal approaches are based on other VSLAM systems, such as *PTAM* [99] and *PoseSLAM* [233]. Moreover, in terms of the objectives

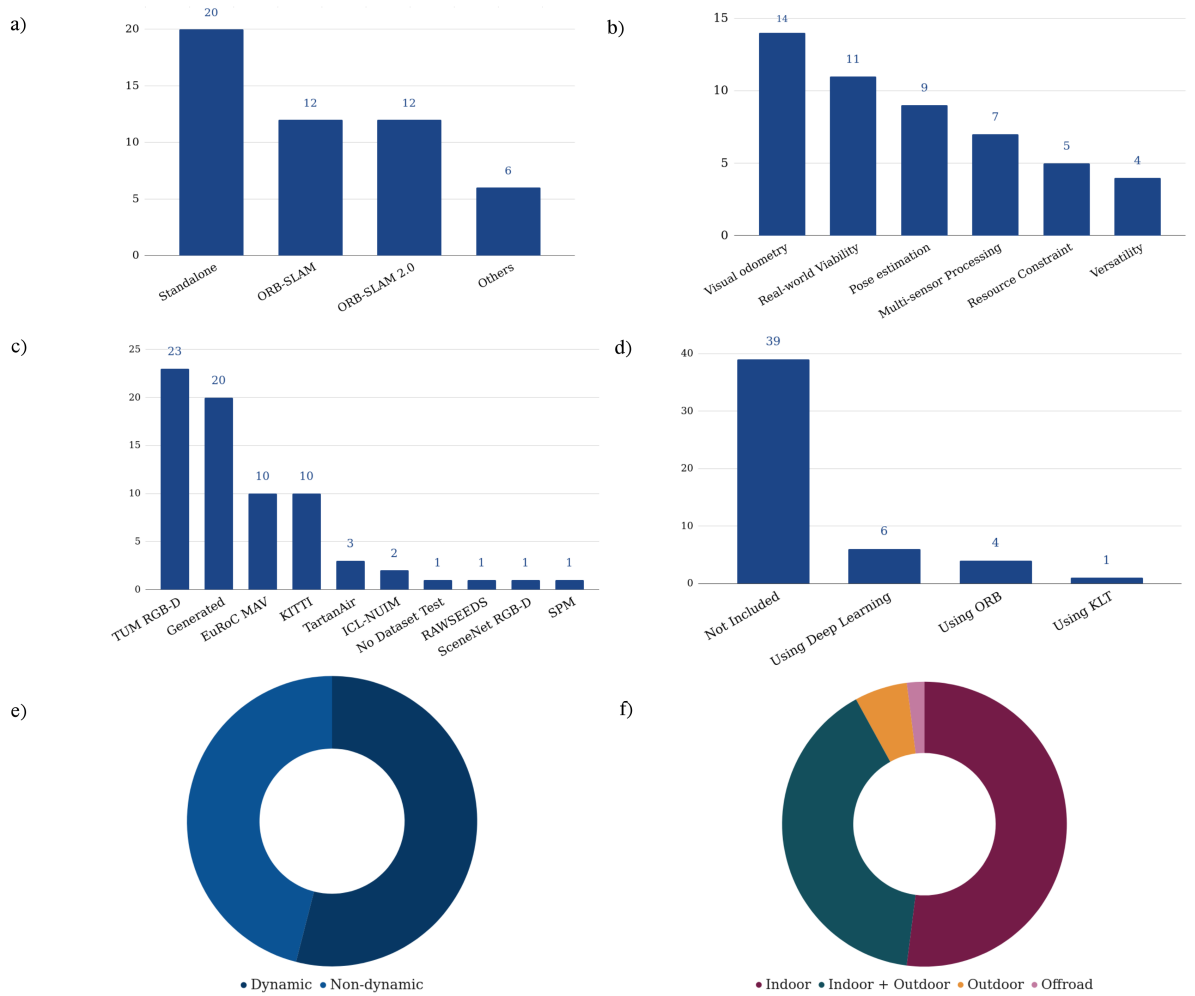


Figure 5.4: Analyzing the current trends of VSLAM approaches: (**a**) base SLAM system employed to implement a new approach, (**b**) the primary objective of the approach, (**c**) various datasets that the proposed methods were tested on, (**d**) the impact of utilizing semantic data in the proposed methods, (**e**) the amount of dynamic objects existing in the environment, (**f**) various types of environments the system tested on.

of VSLAM applications (**RQ2**), what tops the chart in Fig. 5.4b is improving the *VO* module. Thus, most of the recent VSLAMs are trying to resolve the problems of current algorithms in determining the position and orientation of the robots. *Real-world viability* and *pose estimation* are further fundamental objectives of proposing new VSLAM papers. Concerning the dataset used for evaluation in the surveyed papers (**RQ3**), Fig. 5.4c illustrates that most works have been tested on the *TUM RGB-D* dataset [167]. This dataset has been employed as the primary or one of the multiple baselines for evaluation in the reviewed manuscripts. Additionally, many researchers tend to perform experiments on the datasets *generated* by them. It is assumed that the primary motivation for generating a dataset is to show how a VSLAM method works in real-world scenarios and if it can be used as an end-to-end application. *EuRoC MAV* [204] and *KITTI* [185] are the next most popular datasets for evaluation in VSLAM works, respectively.

Another interesting piece of information extracted from Fig. 5.4d concerns the impact of employing semantic data when using the VSLAM system. It can be seen that the majority of the reviewed papers do not include semantic data while processing the

environment. The reason behind not utilizing semantic data could be due to the below reasons:

- The computational cost for training a model that recognizes objects and utilizing it for semantic segmentation is considerable in many cases, which might raise the processing time.
- The majority of the geometry-based VSLAM works are designed to work as plug-and-play devices so that they can employ camera data for localization and mapping with the least possible effort.
- Incorrect information extracted from the scene can also lead to adding more noise to the process.

When the environment is considered, it can be seen in Fig. 5.4e, that more than half of the approaches can also work in dynamic environments with challenging conditions, while the remaining systems are only focused on an environment with no dynamic changes. Moreover, in Fig. 5.4f, most of the approaches work in "indoor settings" or "both indoor and outdoor environments", while the rest of the papers have only been tested in outdoor conditions. It should be mentioned that approaches that can only work in a particular circumstance with restricted assumptions might not produce the same accuracy if employed in other scenarios. That is one of the main reasons why some approaches only concentrate on a particular situation.

5.7.2 Analyzing the Current Trends

The current survey has reviewed the state-of-the-art visual SLAM approaches that have attracted massive attention and demonstrated their principal contributions to this field. Despite the wide range of reliable solutions and improvements in various modules of VSLAM systems over the past years, there are still many high-potential fields, and unsolved issues that are investigating them may lead to more robust approaches in the future evolution of SLAMs. In light of the wide range of visual SLAM approaches, the authors propose currently trending areas for investment and introduce the following open research directions:

I. Various Feature Processing

Deep neural networks have shown encouraging results in various applications, including VSLAM [96], making them a significant trend in multiple fields of study. Due to their learning capabilities, these architectures have shown considerable potential to be utilized as reliable feature extractors to tackle different issues in VO and loop closure detection. CNNs can aid VSLAMs in precise object detection and semantic segmentation and can outperform traditional feature extraction and matching algorithms for correctly recognizing hand-crafted features. It has to be mentioned that since deep learning-based methods have been trained on datasets with large amounts of diversified data and limited object classes, there is always a risk of misclassification of dynamic points, causing false segmentation.

II. Information Retrieval and Computational Cost Trade-Off

Generally, the processing cost and the quantity of information in the scene should always be balanced. In this perspective, dense maps allow VSLAM applications to record high-dimensional complete scene information, but doing so in real-time would be computationally demanding. Sparse representations, on the other hand, would fail to capture

all the needed information due to their lower computational cost. It should also be noted that real-time performance is directly related to the camera's frame rate, and frame losses in peak processing times can negatively affect the VSLAM system's performance, regardless of algorithm performance. Moreover, VSLAMs typically take advantage of tightly coupled modules and modifying one module may adversely affect others, which makes the balancing task more challenging.

III. Semantic Segmentation

Providing semantic information while creating the environment map can bring about beneficial information for the robot. Identifying objects in the camera's field of view—*e.g.,* doors, windows, people, *etc.*—is a trendy topic in current and future VSLAM works, as the semantic information can be used in pose estimation, trajectory planning, and loop closure detection modules. In this regard, 3D LiDAR-based frameworks, such as Situational Graphs (*S-Graphs*) [155], employ planar surfaces and semantic data to illustrate the surroundings face trouble in areas with a high presence of glass. With the widespread usage of object detection and tracking algorithms, semantic VSLAMs will undoubtedly be among the future solutions in this domain.

IV. Loop Closing Algorithms

One of the critical issues in any SLAM system is the drift problem and losing the feature tracks caused by the accumulated localization errors. Detection of drifts and loop closures to identify previously visited places contributes to high computation latency and cost in VSLAM systems [143]. The main reason is that the complexity of loop closure detection increases with the size of the reconstructed map. Moreover, combining the map data collected from various locations and refining the estimated poses are very complex tasks. With this, optimization and balancing of the loop closure detection module have massive potential for improvement. One of the common approaches for detecting loop closures is improving image retrieval by training a visual vocabulary based on local features and then aggregating them.

V. Working in Challenging Scenarios

Working in a texture-less environment with few salient feature points often leads to drift errors in position and orientation in robots. As one of the primary challenges in VSLAM, this error may lead to system failure. Thus, considering complementary scene-understanding methods in feature-based approaches, such as object detection or line features, would be a trendy topic.

5.8 Conclusions

This paper presented a broad range of SLAM works equipped with visual sensors to collect data, known as Visual SLAM (VSLAM). It investigated 50 state-of-the-art approaches and 10+ survey works published in the VSLAM domain. The paper studied the evolution of VSLAM algorithms and identified the highly mature frameworks that most subsequent work is built upon. Additionally, various setups concerning data acquisition means, visual features, target environments, technique evaluations, and semantic levels are addressed in this manuscript. The foremost contribution of this article, compared to previous survey publications, is its categorization of methodologies by their objectives for proposing new solutions. Moreover, recent directions in VSLAM and well-known evaluation criteria for test environments and benchmarks were other fundamental topics covered in this

paper. The essay also introduced existing research gaps and potential contributions that researchers can pursue based on current trends in the VSLAM domain.

6

Marker-Based Visual SLAM

The third paper [166] was accepted at **IROS'23** and presented in December 2023 in Detroit, MI, USA. The work explores the potential of integrating fiducial markers into Visual SLAM to enhance the robot's understanding of its surrounding environment. It laid the groundwork for developing a more general VSLAM framework enriched with environment-driven 3D scene graphs.

Paper Card

Title: Marker-Based Visual SLAM Leveraging Hierarchical Representations
Authors: A. Tourani, H. Bavle, J.L. Sanchez-Lopez, R. Muñoz-Salinas, and H. Voos
Venue: 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
DOI: https://doi.org/10.1109/IROS55552.2023.10341891
Video: https://youtu.be/Dwh2lGLinyo?si=3VRdTWUUY32aTmT4

Copyright

© 2026 by the authors. This article is an open-access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/.)

6.1 Abstract

Fiducial markers can encode rich information about the environment and aid Visual SLAM (VSLAM) approaches in reconstructing maps with practical semantic information. Current marker-based VSLAM approaches mainly utilize markers for improving feature detections in low-feature environments and/or incorporating loop closure constraints, generating only low-level geometric maps of the environment, prone to inaccuracies in complex environments. To bridge this gap, this paper presents a VSLAM approach utilizing a monocular camera along with fiducial markers to generate hierarchical representations of the environment while improving the camera pose estimate. The proposed approach detects semantic entities from the surroundings, including walls, corridors, and rooms encoded within markers, and appropriately adds topological constraints among them. Experimental results on a real-world dataset collected with a robot demonstrate that the proposed approach outperforms a marker-based VSLAM baseline in terms of accuracy, given the addition of new constraints while creating enhanced map representations. Furthermore, it shows satisfactory results when comparing the reconstructed map quality to that of the one rebuilt using a LiDAR SLAM approach.

6.2 Introduction

VSLAM systems can employ a wide range of vision sensors, such as monocular, stereo, Red Green Blue-Depth (RGB-D), omnidirectional, and event-based cameras to estimate the environmental map while localizing the camera [39]. The primary advantage of vision sensors is that they need low-cost hardware to supply rich visual and semantic information from the surroundings for various tasks [47]. Using such sensors, semantic data, which refers to high-level information acquired from the environment, makes VSLAM tasks more robust and expands the range of applications that can employ the reconstructed maps [39, 237]. For instance, robots may need to identify objects in the scene for path planning and dynamic object removal, which can be resolved using semantic data [209]. In this regard, utilizing fiducial markers is one of the possible approaches for encoding semantic information into the environment due to their rich and better-defined features [238]. In addition, they can assist VSLAM frameworks by providing accurate pose estimation due to their unique patterns, supplying reliable features in low-texture environments, and enabling loop closure detection based on marker identifiers. While most fiducial markers are visible to humans and, thus, visually pollute the environment, we have recently proposed novel *invisible fiducial markers* [12] that have the potential to be seamlessly integrated into the environment. Nevertheless, for the sake of simplicity, in this work, we use visible fiducial markers, leaving the use of *invisible markers* as future work.

Recent works such as [150] and [152] propose VSLAM approaches using fiducial markers but do not encode them with meaningful semantic information, creating purely geometric map representations, leading to inaccuracies in camera pose estimates and the generated map in the presence of complex environments. To fully leverage the potential of fiducial markers in accurately identifying both the semantic elements and their topological relationships while generating accurate and meaningful maps, this paper proposes a VSLAM framework for monocular cameras that utilizes the data encoded in fiducial markers for enhanced map reconstruction, adding abstract semantic elements to the final map along with their topological relationships. The system is built upon UcoSLAM [150], which employs ArUco [82] markers and visual keypoints extracted from natural landmarks, reconstructing the geometric map of the environment using keypoints and

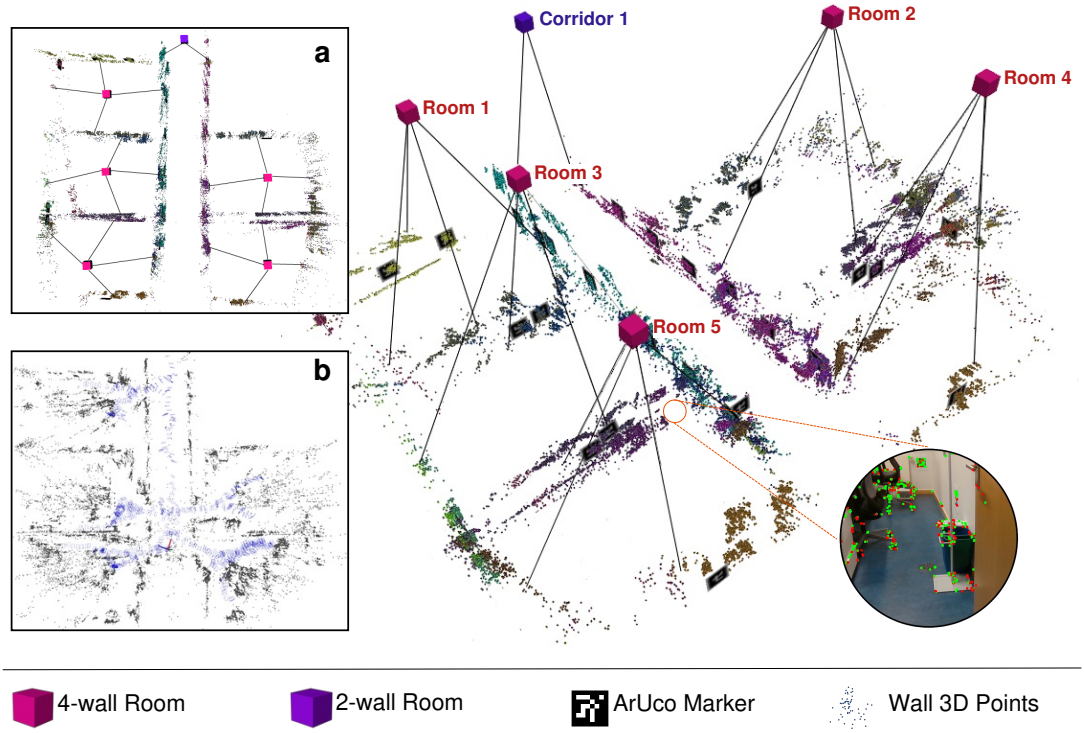


Figure 6.1: The final reconstructed map of the environment using the proposed method in a hierarchical representation. Accordingly, the primary map entities are detected as 3D points extracted from the environment and visited ArUco markers. Various colors of the 3D points refer to distinct walls, and the line connecting them to the cubes shows their belonging to rooms, including corridors or four-wall rooms: **a)** the top view of the reconstructed map represented in 2D, **b)** keypoints and robot trajectory records that lead to map reconstruction.

markers. Additionally, inspired by *S-Graphs* [155, 156], leveraging high-level hierarchical representations for localization and mapping, the proposed approach adds extraordinary information in the form of walls, corridors, and rooms, considering the data encoded in the ArUco markers attached to the environment. A sample map reconstructed by the proposed approach and its hierarchical representation is demonstrated in Fig. 6.1.

Herewith, the main contributions of the paper are summarized below:

- An extension of a marker-based VSLAM framework to reconstruct environment maps with high-level semantic features, including walls, corridors, and rooms, with the aid of fiducial markers,
- The design of novel geometric constraints, namely *marker-to-wall* and *wall-to-room*, to improve the map quality and reduce camera localization errors,
- Validation of the proposed method using a real-world indoor dataset showing improved performance over the marker-based VSLAM baseline.

The rest of the paper is organized as follows: Section 6.3 explores previous efforts in the VSLAM domain, targeting the usage of semantic data for improved reconstructed maps. Section 6.4 details the proposed approach and its various modules. In Section 6.5, the evaluation criteria and discussions over the effectiveness of the proposed method are provided. The paper finally concludes in Section 6.6.

6.3 Related Works

VSLAM approaches have matured over the past years, and many researchers have contributed to this community by proposing efficient solutions to unresolved challenges in this area. The authors of this paper also presented a comprehensive survey on diverse VSLAM state-of-the-art works and studied their trade-offs in [166]. According to the mentioned survey, many recent approaches focus on improving the *front-end* thread of the VSLAM systems for better pose estimation and visual odometry [182]. Authors in [150] proposed UcoSLAM that employs both natural and artificial landmarks (*i.e.,* fiducial markers) in the environment to reconstruct the map. It works in keypoints-only, markers-only, and mixed modes and improves the global map builder and loop closure detector modules in case ArUco markers are found in the camera's field of view. Another marker-based approach titled TagSLAM [152] was proposed by Pfrommer and Daniilidis, which uses the capability of AprilTag [81] fiducial markers for SLAM tasks. Although the proposed approaches perform well in many scenarios, they cannot obtain semantic information from the environment for an improved map representation. Gomez-Ojeda *et al.* [110] introduced PL-SLAM, which uses Oriented FAST and Rotated BRIEF (ORB) and Line Segment Detector (LSD) algorithms to extract visual features of points and lines in stereo vision cameras. However, this approach is computationally intensive in feature tracking and extracting data from structural lines. Bruno and Colombini [117] proposed LIFT-SLAM, which integrates deep learning-based and conventional geometry-based feature descriptors. Despite the robust feature detection and tracking performance, the method uses an unoptimized Convolutional Neural Network (CNN), which leads to almost real-time execution.

Utilizing semantic data is another topic that attracted significant attention in recent years. Gonzalez *et al.* [239] presented S^3LAM, a monocular approach based on ORB-SLAM 2.0 [107] that uses semantic segmentation of generic objects and structures. Their approach fits a plane using the triangulated 3D points for each a priori planar cluster and uses a CNN-based panoptic segmentation framework titled *Detectron2* [240]. Likewise, YOLO-SLAM [241] is another CNN-based approach that couples geometric constraints with the Darknet19-YOLOv3 object detector to generate semantic information. However, these solutions may suffer from performance degradation in recognizing small or non-regular objects and planes. QuadricSLAM [242] is another approach that models objects found in the environment by employing quadric representations as object descriptions and estimates quadric parameters using object detection bounding boxes. This monocular VSLAM represents an object's size, position, and orientation for map reconstruction. The main challenge of QuadricSLAM is the initialization stage, where a variety of viewing angles of the object is required. Hosseinzadeh *et al.* [243] proposed another monocular VSLAM system that inserts visited objects detected by a deep learning-based object detector and the dominant structure of the scene using a planar landmark detector to the map. Despite the robust performance, estimating the prior shape of the objects leads to high computational complexity. Another semantic VSLAM approach titled Blitz-SLAM [244] combines the original masks and depth information of the objects and eliminates noise blocks formed by dynamic objects. Authors in [245] proposed a parallel framework titled SOLO-SLAM, which uses semantic attributes of map points and geometric constraints to filter dynamic objects and reconstruct an improved map. The main challenge here is losing much information in low-dynamic scenarios. MISD-SLAM [246] is another semantic approach equipped with real-time instance segmentation based on a pre-trained

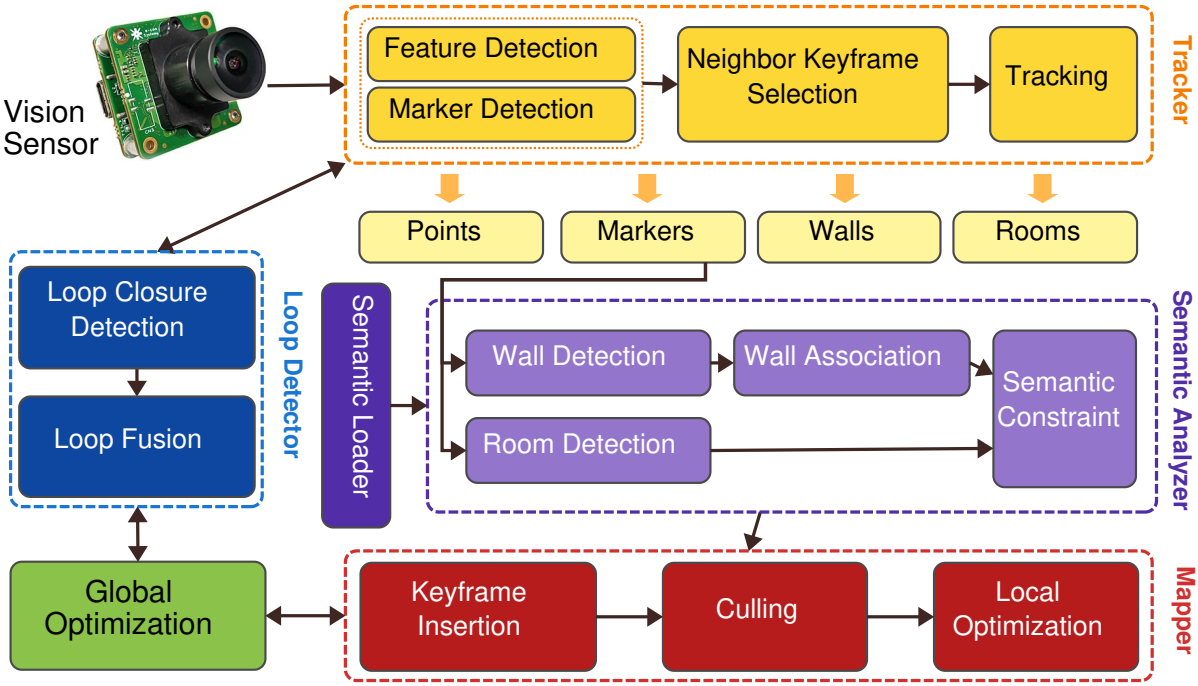


Figure 6.2: The overall pipeline of the proposed approach. The main contributions of the proposed method are adding the *Semantic Analyzer* module and modifying the output of the *Tracker* to provide proper feed to it. Moreover, the *Global/Local Optimizer* modules have also been modified for better optimization and map reconstruction.

CNN and dynamic feature removal. However, the algorithm removes static objects only in some frames.

In contrast with the mentioned works, the proposed approach creates a map of the environment using semantic data encoded within makers instead of employing any object detector. Accordingly, walls and room constraints are estimated based on mathematical modeling obtained from keypoint and marker data, which does not significantly affect the system's performance. Additionally, the accuracy of the final reconstructed map of the environment is comparable with that of the one from Light Detection And Ranging (LiDAR)-based approaches, demonstrating the proposed method's potential.

6.4 Proposed Method

The pipeline of the proposed method is depicted in Fig. 6.2. Accordingly, the frames captured by a monocular camera are passed to the system for processing. The *tracker* module detects ORB visual features and ArUco markers as the primary inputs of the system. Keyframes are selected and tracked over time, and the output contains 3D map points, detected markers, walls, and rooms visited in the scene. The *semantic analyzer* module performs wall and room detection tasks using the pose of the markers detected in the *tracking* stage. Additionally, a *mapper* module handles adding new vertices and edges to the map and improves the final map for an optimized reconstruction. Finally, a marker-based *loop detector* module checks whether the markers in the current frame have been seen before and triggers the *global optimizer* to enhance the reconstructed map.

In order to develop a VSLAM framework with richer reconstructed maps, the proposed approach utilizes UcoSLAM as the baseline and modifies its components to be

empowered with a semantic data analysis procedure. The mentioned modifications enable the detection of *walls* and different types of *rooms* as two semantic concepts in the environment using ArUco markers. Furthermore, as these fiducial markers are used for camera pose estimation, they can store additional encoded semantic data, which makes them an ideal alternative according to their unique identification properties. This method also aspires to employ visual data to represent the environment and the robot's pose in a single optimizable graph with an approach comparable to *S-Graphs* [155]. Thus, instead of odometry readings and extracting planar surfaces using LiDAR scans introduced by *S-Graphs* and *S-Graphs+* [156], the proposed approach calculates the location of walls and room variations using the fiducial markers attached to them.

6.4.1 Overview

The pipeline of the proposed method at time t can be referred to four main coordinate systems: the odometry frame of reference O, the camera coordinate system C_t, the marker coordinate system M_t, and the global coordinate system G_t. As the primary sensor of the system, a monocular camera acquires a set $\mathbf{F} = \{\mathbf{f}\}$ of frames $\mathbf{f} = \{t, \mathbf{T}, \delta\}$, where $\mathbf{T} \in SE(3)$ is the camera's pose obtained from the transformation of C_t to G_t, and δ is the set of camera intrinsic parameters. By processing each camera frame using ORB keypoint detector and feature extractor, a set of keypoints $\mathbf{g} = \{l, u, \mathbf{d}\}$ are extracted in which l is the sub-sampling level of the image, u is the pixel coordinates for up-sampling w.r.t. the first level, and $\mathbf{d} = (d_1...d_n)|d_i \in [0, 1]$ is the descriptor vector with length n. Accordingly, the final constructed map of the environment $\mathbf{E}$ will be represented as:

$$\mathbf{E} = \{\mathbf{K}, \mathbf{P}, \mathbf{M}, \mathbf{W}, \mathbf{R}\} \tag{6.1}$$

where $\mathbf{K} = \{k\} \subset \mathbf{F}$ is the set of keyframes and $\mathbf{P} = \{\mathbf{p}\}$ represents the set of feature points $\mathbf{p} = \{\mathbf{x}, \mathbf{v}, \hat{\mathbf{d}}\}$ extracted from the environment with their corresponding 3D positions $\mathbf{x} \in \mathbb{R}^3$, viewing direction $\mathbf{v} \in \mathbb{R}^3$, and descriptor $\hat{\mathbf{d}}$. Additionally, $\mathbf{M} = \{\mathbf{m}\}$ is the set of ArUco markers detected in the environment, in which each marker $\mathbf{m} = \{s, \mathbf{p}, \mathbf{cc}\}$ holds marker size (*i.e.,* length) $s \in \mathbb{R}$, marker pose $\mathbf{p} \in SE(3)$ calculated from M_t to G_t, and marker corner coordinates $\mathbf{cc} = (c_1...c_4)|c_i \in \mathbb{R}^3$ values. The set of walls detected from the environment is represented by $\mathbf{W} = \{\mathbf{w}\}$, in which each wall $\mathbf{w} = \{\mathbf{q}, \mathbf{m_w}\}$ holds the wall equation $\mathbf{q} \in \mathbb{R}^4$ and the attached markers list $\mathbf{m_w} \subset \mathbf{M} = (m_1...m_n)|m_i \in \mathbb{N}$ where m_i represents ArUco *marker-id*. Similarly, $\mathbf{R} = \{\mathbf{r}\}$ refers to the set of rooms found in the environment, where each room $\mathbf{r} = \{\mathbf{r_c}, \mathbf{r_w}\}$ contains the room center point $\mathbf{r_c} \in \mathbb{R}^3$ and the wall list $\mathbf{r_w} \subset \mathbf{W} = (w_1...w_n)|w_i \in \mathbb{N}$ that comprise the room.

6.4.2 Semantic Entities

The proposed approach adds two semantic entities to construct a richer map, including *walls* and *rooms* (containing two-wall and four-wall rooms). As detecting the mentioned entities is accomplished differently, the procedures are discussed in detail in this section.

Walls. The procedure to find the walls as plane-shaped entities on which the markers and their surrounding features are located is predominantly done using the pose information provided by detected ArUco markers. This approach assumes all fiducial markers are placed directly on the walls, not on other elements such as radiators or drawer units. Thus, the equations of the walls are obtained based on the attached markers' poses. The process of detecting walls in the scene occurs whenever a marker is being visited in a recent keyframe, making detecting and mapping the walls at a given time more efficient.

To add walls to the final map, each wall plane $\boldsymbol{w}_i$ is extracted in the global coordinate system ${}^{G}\boldsymbol{w}_i = \begin{bmatrix} {}^{G}\boldsymbol{n}_i & {}^{G}d \end{bmatrix}$ with a normal vector ${}^{G}\boldsymbol{n}_i = \begin{bmatrix} n_x & n_y & n_z \end{bmatrix}^T$. The vertex node of the wall is factored in the graph as ${}^{G}\boldsymbol{w_i} = [{}^{G}\phi, {}^{G}\theta, {}^{G}d]$, where ${}^{G}\phi$ and ${}^{G}\theta$ refer to the azimuth and elevation of the wall in G_t. For each marker ${}^{G}\boldsymbol{m}_i$ attached to the wall ${}^{G}\boldsymbol{w}_i$ cost function can be defined as:

$$c_{w_i}({}^{G}\boldsymbol{w}_i, {}^{G}\boldsymbol{m}_i) = \|[{}^{M}\delta\phi_{w_{i m_i}}, {}^{M}\delta\theta_{w_{i m_i}}, {}^{M}d_{w_i}]^T\|^2_{\boldsymbol{\Lambda}_{\tilde{\boldsymbol{w}}_i}} \tag{6.2}$$

where ${}^{M}\delta\phi_{w_{i m_i}}$ difference between the azimuth angle of the wall w_i and its marker m_i converted to its marker frame M_i, ${}^{M}\delta\theta_{w_{i m_i}}$ is the difference in the elevation angles, while ${}^{M}d_{w_i}$ being the perpendicular distance between the wall and the marker, which should be zero the given marker-wall pair.

Rooms. Another semantic entity considered in this work is the room. Since perceiving a room can be difficult due to various configurations and structures, the proposed approach employs the data encoded in ArUco markers attached to the room's walls to detect the mentioned semantic entity. In this regard, a dictionary containing the rooms in the environment and the fiducial markers attached to their walls is provided to feed the framework (*Semantic Loader* in Fig. 6.2). It should be noted that the only information encoded in the dictionary is the *marker-ids* corresponding to a room, and no additional pose information is required to be encoded. The proposed approach presents a modified version of the room segmentation process introduced in *S-Graphs+* [156], where the markers play a vital role. Hence, two room variations titled "*two-wall room*" and "*four-wall room*" have been considered in this work:

***I. Two-wall Rooms (Corridors)*:** In this case, only two parallel walls of a room are labeled with fiducial markers. This scenario is proper for detecting corridors or rooms with undetectable/unreachable walls in the scene. Consequently, a room ${}^{G}\mathbf{r}_x = [{}^{G}\mathbf{w}_{x_{a_1}}, {}^{G}\mathbf{w}_{x_{b_1}}]$ contains x-wall planes parallel to the x-axis while ${}^{G}\mathbf{r}_y = [{}^{G}\mathbf{w}_{y_{a_1}}, {}^{G}\mathbf{w}_{y_{b_1}}]$ contains y-wall planes parallel planes to y-axis. To compute the center of a two-wall room ${}^{G}\mathbf{r}_{x_i}$, the two x-wall plane equations are utilized along with the center point ${}^{G}\mathbf{c}_i$ of the marker $\mathbf{m}_i$ as follows:

$$\begin{aligned} {}^{G}\mathbf{k}_{x_i} &= \frac{1}{2}\big[|{}^{G}d_{x_{a_1}}| \cdot {}^{G}\mathbf{n}_{x_{a_1}} - |{}^{G}d_{x_{b_1}}| \cdot {}^{G}\mathbf{n}_{x_{b_1}}\big] + |{}^{G}d_{x_{b_1}}| \cdot {}^{G}\mathbf{n}_{x_{b_1}} \\ {}^{G}\boldsymbol{\eta}_{x_i} &= {}^{G}\hat{\mathbf{k}}_{x_i} + \big[{}^{G}\mathbf{c}_i - [\ {}^{G}\mathbf{c}_i \cdot {}^{G}\hat{\mathbf{k}}_{x_i}]\ \cdot {}^{G}\hat{\mathbf{k}}_{x_i}\big] \end{aligned} \tag{6.3}$$

where ${}^{G}\boldsymbol{\eta}_{x_i}$ is the center point of the two-wall room ${}^{G}\mathbf{r}_{x_i}$ and ${}^{G}\hat{\mathbf{k}}_{x_i}$ is acquired from ${}^{G}\hat{\mathbf{k}}_{x_i} = {}^{G}\mathbf{k}_{x_i}/\|{}^{G}\mathbf{k}_{x_i}\|$. The center point ${}^{G}\mathbf{c}_i$ of the marker is obtained using the marker pose in frame G. It should be noted that a two-wall room center in the y direction can be calculated analogously.

A two-wall room node is initialized using the room center, and the cost function to minimize the two-wall room node and its corresponding wall planes is defined below:

$$c_{\boldsymbol{r}_{x_i}}({}^{G}\boldsymbol{r}_{x_i}, \big[{}^{G}\mathbf{w}_{x_{a_1}}, {}^{G}\mathbf{w}_{x_{b_1}}, {}^{G}\mathbf{c}_i\big]) \quad = \quad \sum_{t=1,i=1}^{T,K} \|{}^{G}\hat{\boldsymbol{\eta}}_{x_i} \ - \ f({}^{G}\tilde{\mathbf{w}}_{x_{a_1}}, {}^{G}\tilde{\mathbf{w}}_{x_{b_1}}, {}^{G}\mathbf{c}_i)\|^2_{\boldsymbol{\Lambda}_{\tilde{\mathbf{r}}_{i,t}}} \tag{6.4}$$

where $f({}^{G}\tilde{\mathbf{w}}_{x_{a_1}}, {}^{G}\tilde{\mathbf{w}}_{x_{b_1}}, {}^{G}\mathbf{c}_i)$ maps the wall planes to the center point of the room using Eq. 6.3.

II. Four-wall Rooms: This scenario refers to rooms with four walls labeled with fiducial markers. In this regard, a four-wall room contains four wall planes as ${}^{G}\mathbf{r}_i =$

$[{}^{G}\mathbf{w}_{x_{a_1}}\,{}^{G}\mathbf{w}_{x_{b_1}}\,{}^{G}\mathbf{w}_{y_{a_1}}\,{}^{G}\mathbf{w}_{y_{b_1}}]$ forming the room. The center point of this variant of rooms can be computed using the equation below:

$$
\begin{aligned}
{}^{G}\mathbf{q}_{x_i} &= \frac{1}{2}\big[|{}^{G}d_{x_{a_1}}| \cdot {}^{G}\mathbf{n}_{x_{a_1}} - |{}^{G}d_{x_{b_1}}| \cdot {}^{G}\mathbf{n}_{x_{b_1}}\big] + |{}^{G}d_{x_{b_1}}| \cdot {}^{G}\mathbf{n}_{x_{b_1}} \\
{}^{G}\mathbf{q}_{y_i} &= \frac{1}{2}\big[|{}^{G}d_{y_{a_1}}| \cdot {}^{G}\mathbf{n}_{y_{a_1}} - |{}^{G}d_{y_{b_1}}| \cdot {}^{G}\mathbf{n}_{y_{b_1}}\big] + |{}^{G}d_{y_{b_1}}| \cdot {}^{G}\mathbf{n}_{y_{b_1}} \\
{}^{G}\boldsymbol{\rho}_i &= {}^{G}\boldsymbol{q}_{x_i} + {}^{G}\boldsymbol{q}_{y_i}
\end{aligned} \tag{6.5}
$$

where ${}^{G}\boldsymbol{\rho}_i$ is the center point of the four-wall room ${}^{G}\mathbf{r}_i$. It should also be noted that Eq. 6.5 holds true when $|d_{x_1}| > |d_{x_2}|$. The cost function to minimize four-wall room nodes and their corresponding wall plane set is similar to a two-wall room, but with minor differences:

$$
\begin{aligned}
&c_{\boldsymbol{\rho}}({}^{G}\boldsymbol{\rho}, \big[{}^{G}\mathbf{w}_{x_{a_i}}, {}^{G}\mathbf{w}_{x_{b_i}}, {}^{G}\mathbf{w}_{y_{a_i}}, {}^{G}\mathbf{w}_{y_{b_i}}\big]) \\
&\qquad = \sum_{t=1,i=1}^{T,S} \|{}^{G}\hat{\boldsymbol{\rho}}_i - f({}^{G}\tilde{\mathbf{w}}_{x_{a_i}}, {}^{G}\tilde{\mathbf{w}}_{x_{b_i}}, {}^{G}\tilde{\mathbf{w}}_{y_{a_i}}, {}^{G}\tilde{\mathbf{w}}_{y_{b_i}})\|^2_{\boldsymbol{\Lambda}_{\tilde{\boldsymbol{\rho}}_{i,t}}}
\end{aligned} \tag{6.6}
$$

where $f({}^{G}\tilde{\mathbf{w}}_{x_{a_i}}, {}^{G}\tilde{\mathbf{w}}_{x_{b_i}}, {}^{G}\tilde{\mathbf{w}}_{y_{a_i}}, {}^{G}\tilde{\mathbf{w}}_{y_{b_i}})$ maps the four estimated wall planes to the center point of the four-wall room using Eq. 6.5. The above cost function maintains structural consistency among the four walls that form the room.

6.4.3 Final Graph

Fig. 6.3 depicts the structure of the final semantic graph produced by the proposed approach. Accordingly, the keyframes extracted by the system are the primary sources of information that contain both visual feature points with their corresponding 3D coordinates and visited ArUco markers in the scene. It should be noted that the constraint defined among the keyframes, feature points, and fiducial markers is utilized for computing the odometry and detecting loop closure. Additionally, the wall-marker constraints are incorporated using Eq. 6.2. The topmost level of the graph retains rooms detected in the environment and the walls that hold those markers with constraints obtained following Eq. 6.4 for two-wall rooms and Eq. 6.6 for four-wall rooms.

6.5 Evaluation

This section presents the proposed method's evaluation procedure regarding accuracy and performance compared to the state-of-the-art approaches. In this respect, various real-world scenario tests were performed using the proposed method, UcoSLAM [150] as the baseline methodology, and *S-Graph+* [156] as a LiDAR-based framework for providing ground truth measurements. As the LiDAR sensor has less noise when compared with the monocular vision-based sensor, it can be safely considered as ground truth.

6.5.1 Evaluation Setup

In order to evaluate the performance of the proposed approach in real-world circumstances, we mounted an *Intel RealSense Depth Camera D435* as the monocular sensor on a *Boston Dynamics Spot* robot and collected data from an indoor environment. The robot functioned in different office zones of two different university buildings with various corridor and room setups, where the walls were labeled with printed $17cm \times 17cm$ ArUco markers. *Marker-ids* of the ArUco markers placed in the environment, along with the

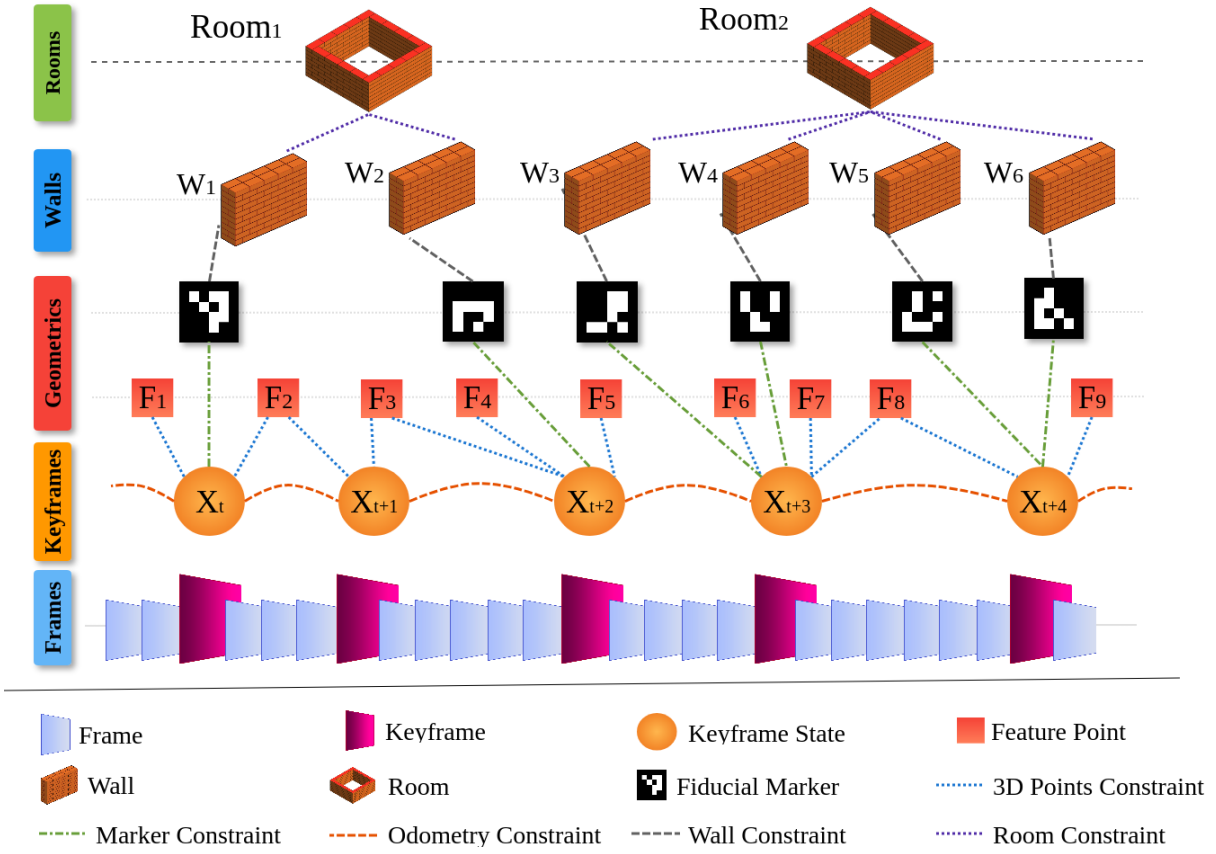


Figure 6.3: The graph representation of the hierarchical architecture of the proposed approach. In this regard, the map employs new semantic constraints, including walls and rooms, along with the existing geometrical constraints for better map reconstruction.

room unique labels, were also stored in a database file and fed to the system as the Semantic Loader module. Fig. 6.4 demonstrates the robot employed for collecting data and some instances of the collected dataset. Additionally, the characteristics of the dataset are presented in Table 6.1.

Evaluations have been conducted using a computer equipped with an *11^{th} Gen. Intel Core i9 @2.60GHz* processor and 32 GigaBytes of memory. Moreover, to better visualize the detected objects in the environment, including 3D points, ArUco markers, walls, and rooms, and provide a more accurate evaluation to compare the proposed method with other state-of-the-art approaches, a set of ROS tools has been employed.

6.5.2 Experimental Results

To demonstrate the accuracy of the proposed method compared to its baseline and ground truth, Absolute Trajectory Error (ATE) measurements have been employed in this paper. Accordingly, the Root Mean Square Deviation (RMSE) and Standard Deviation (STD) values of the proposed and baseline approaches were compared to the ground truth, and the process with the lower value is assumed to perform more accurately.

According to the evaluation results presented in Table 6.2, the proposed approach works better than its baseline in most cases. The main reason for such improvement is the ability of the proposed method to add new constraints to the map and employ the association of semantic entities to enhance the reconstruction of the final map. The mentioned improvement can also be seen in Fig. 6.6. Moreover, the proposed approach



Figure 6.4: Collected dataset for evaluation of the method: **a)** the legged robot used for data collection, **b)** some instances of the dataset.

Table 6.1: The characteristics of the collected indoor dataset.

Seq.*	***Duration***	***#Markers***	***Description***
Seq-01	08m 53s	35	Two corridors connected via a landing (without robot rotation)
Seq-02	09m 29s	35	Two corridors connected via a landing (with robot rotation)
Seq-03	16m 05s	26	One corridor and five rooms
Seq-04	10m 25s	14	One corridor and a two-doors room
Seq-05	09m 37s	20	Four aisles connected to a corridor
Seq-06	23m 34s	22	Three corridors and three rooms

*data were stored as packages of *rosbag* files.

utilizes data association results and performs a local *semantic loop closure* in addition to the marker-based loop closure detection when visiting a previously observed wall. In this regard, if the equation of the currently visible wall was previously measured, *i.e.,* the wall was previously detected, a global optimization runs again to improve the reconstructed map.

The impact mentioned above is more evident in cases such as *Seq-04* where the robot starts in a corridor, enters from one of the two doors of a room, and exits from the other door to continue in the same corridor, and as a result, no loop closure using keypoints and markers is performed. While our method, which creates wall and room constraints, correctly identifies the corridor and its walls to provide better results. This improvement can be seen in Fig. 6.5, where our method can reconstruct a more accurate map of the environment when compared with the baseline. Moreover, it can extract meaningful semantic and topological information from the environment.

Limitations. The proposed method, when testing in an environment where marker-based loop closure is triggered several times, can show similar results to baseline UcoSLAM. *Seq-06* in Fig. 6.5 shows an example of the final map showcasing similar map quality results generated by the proposed method and UcoSLAM for an environment where the robot revisits the places (thus the markers) several times. Even with the given limita-

Table 6.2: Evaluation results of the proposed method on the collected dataset using Root Mean Square Deviation (RMSE) error in *meters* and Standard Deviation (STD). The best results are boldfaced.

	Method			
	Proposed		UcoSLAM [150]	
Sequence	RMSE	STD	RMSE	STD
Seq-01	**8.038**	3.058	8.130	**3.035**
Seq-02	**6.883**	**3.598**	6.930	3.633
Seq-03	**2.266**	**0.894**	2.687	1.335
Seq-04	**3.787**	**1.848**	5.822	2.726
Seq-05	1.255	0.763	**1.238**	**0.751**
Seq-06	**2.676**	**1.524**	2.720	1.536

tion for such scenarios, our approach can extract more meaningful information from the environment while maintaining similar accuracy w.r.t. the baseline.

6.6 Conclusions

This paper presented a VSLAM approach that employs a monocular camera as the sensor and the data encoded within fiducial markers placed in the surroundings for accurate pose estimation and semantic segmentation. The proposed method can detect three practical semantic entities in the environment, including walls, corridors, and rooms, and provide a hierarchical graph with high-level semantic data. Additionally, it can reconstruct a more accurate map of the environment by adding more topological constraints and employing the relations among the mentioned semantic entities found in the scene. Evaluations performed on a dataset collected by a legged robot in real-world conditions demonstrated an accuracy improvement compared to the baseline and satisfactory results compared to LiDAR-based state-of-the-art techniques.

The introduced method is part of a project that investigates the advantages of fiducial markers for various tasks while keeping these markers invisible to human eyes but recognizable to robots. As for future works, the authors plan to encode the environment with the mentioned markers introduced in [12] and equip the robot with a proper sensor setup to interpret them, making the improved version of the proposed approach able to accomplish Simultaneous Localization and Mapping tasks with the new type of markers. Additionally, improving the performance of the proposed system to work online and in real-time is another target for future work.



(a) *S-Graphs+* on *Seq-04*

(b) UcoSLAM on *Seq-04*

(c) Proposed method on *Seq-04*

(d) Proposed method on *Seq-04*, only wall 3D points



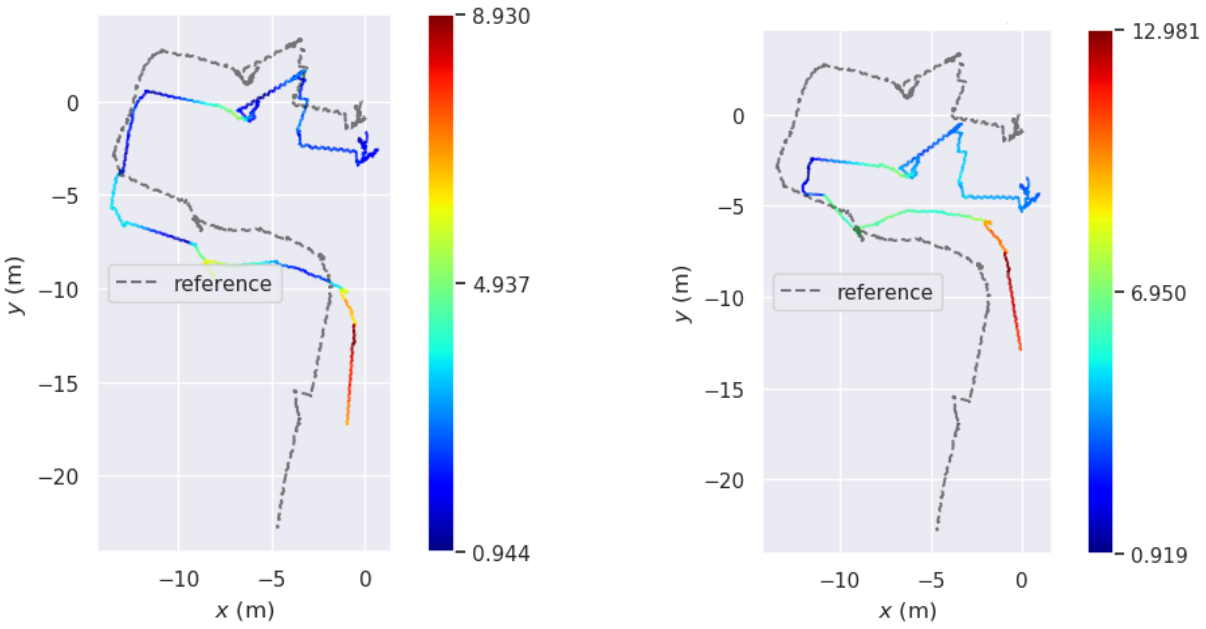
(e) *S-Graphs+* on *Seq-06*

(f) UcoSLAM on *Seq-06*

(g) Proposed method on *Seq-06*

(h) Proposed method on *Seq-06*, only wall 3D points

Figure 6.5: Reconstructed maps using the proposed method, the baseline, and the ground truth. It can be seen that the proposed approach is able to generate more precise maps similar to the ground truth in detail and with high-level semantic entities.



(a) Proposed

(b) UcoSLAM

Figure 6.6: Absolute Trajectory Error (ATE) w.r.t. translation part in *meters* on *Seq-04* using different approaches. It can be seen that the room created by the proposed approach is more similar to the ground truth.

7

Marker-Driven Semantics

The fourth paper [52] was accepted for publication in **MDPI Robotics** in July 2024. This work investigates enhanced marker-driven semantic entity integration within Visual SLAM, establishing a foundation for utilizing fiducial markers in environment understanding and digital twin generation. Notably, it represents the first effort to employ **iMarker** prototypes alongside conventional printed fiducial markers for detecting structural entities in indoor environments.

Paper Card

Title: Vision-Based Situational Graphs Exploiting Fiducial Markers for the Integration of Semantic Entities
Authors: A. Tourani, H. Bavle, D.I. Avşar, J.L. Sanchez-Lopez, R. Muñoz-Salinas, and H. Voos
Venue: MDPI Robotics
DOI: https://doi.org/10.3390/robotics13070106
Video: https://youtu.be/swqvJFJkuMI?si=DKiIjbD4mCCeMFD5

Copyright

© 2026 by the authors. This article is an open-access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/.)

7.1 Abstract

Situational Graphs (S-Graphs) merge geometric models of the environment generated by Simultaneous Localization and Mapping (SLAM) approaches with 3D scene graphs into a multi-layered, jointly optimizable factor graph. As an advantage, S-Graphs not only offer a more comprehensive robotic situational awareness by combining geometric maps with diverse, hierarchically organized semantic entities and their topological relationships within one graph, but they also lead to improved performance of localization and mapping on the SLAM level by exploiting semantic information. In this paper, we introduce a vision-based version of S-Graphs where a conventional Visual SLAM (VSLAM) system is used for low-level feature tracking and mapping. In addition, the framework exploits the potential of fiducial markers (both visible and our recently introduced transparent or fully invisible markers) to encode comprehensive information about environments and the objects within them. The markers aid in identifying and mapping structural-level semantic entities, including walls and doors in the environment, with reliable poses in the global reference, subsequently establishing meaningful associations with higher-level entities, including corridors and rooms. However, in addition to including semantic entities, the semantic and geometric constraints imposed by the fiducial markers are also utilized to improve the reconstructed map's quality and reduce localization errors. Experimental results on a real-world dataset collected using legged robots show that our framework excels in crafting a richer, multi-layered hierarchical map and enhances robot pose accuracy at the same time.

7.2 Introduction

Employing vision sensors for Simultaneous Localization and Mapping (SLAM) applications in mobile robotics can bring about several merits, including the ability to achieve rich visual information using a low-cost hardware setup, making them attractive approaches compared to Light Detection And Ranging (LiDAR)-based tools [180]. These variants of SLAM systems are known as Visual SLAM (VSLAM), as they employ visual data for map reconstruction. However, besides building a geometric map of the environment and being able to localize within it, advanced robotic situational awareness also requires the extraction of more abstract semantic information. To incorporate semantic data, approaches such as [247] enrich VSLAM with high-level information about the environment, but many of these solutions do not yet integrate valuable relational information among pertinent semantic entities. To fill this gap, 3D scene graph methodologies, such as the works introduced in [159, 161, 157], exhibit promising results generating meaningful 3D scene graphs from underlying SLAM data and include dynamic, semantic, and topological relationships of the situation. While these approaches still keep the different graphs separately, the authors recently proposed, as a further improvement, a novel approach called Situational Graphs (S-Graphs) [155], merging SLAM graphs and scene graphs into one multi-layered jointly optimizable factor graph with improved performance.

However, while the S-Graph approach only works with LiDAR sensors so far, this paper introduces a vision-based version by incorporating a VSLAM system. In addition, a vision-based approach also simplifies the detection of particular semantic entities by exploiting visible artificial landmarks. In this regard, fiducial markers, as well-known artificial landmarks for Augmented Reality (AR)/Mixed Reality (MR) or robotic tasks, such as the commonly used AprilTag [81] or ArUco [82] markers, can aid in rapid information decoding and recognition. While these conventional and often paper-based fiducial

markers are visible to the human eye and, hence, often found to be optically distracting in many practical applications, we recently invented iMarkers—fiducial markers made with Cholesteric Spherical Reflectors (CSRs) [12]. The use of CSRs allows for the production of mechanically very robust fiducial markers that are transparent or even shift the band of the reflected light to the UV or IR band, making them invisible to humans but still detectable for robots equipped with a specialized yet simple optical sensor. Therefore, the vision is that iMarkers will allow the future application of fiducial markers in much larger quantities in real environments without any optical distraction for humans, which finally motivates our marker-based approach in this paper. To maintain the paper's focus on robotics and the potential of using iMarkers in visual SLAM, and to avoid going into detail about the markers and their production procedure, the authors invite the readers to explore more detailed explanations and discussions in [12].

Fiducial markers, when employed in VSLAM applications, can help to extract geometric information from the environment, and hence some VSLAM methodologies like UcoSLAM [150] and TagSLAM [152] used this potential of conventional fiducial markers to facilitate the creation of geometric maps. However, in these approaches, the performance of the localization and mapping strongly depends on the availability and detectability of the fiducial markers in the environment. To better exploit the role of fiducial markers in VSLAM, our previous preliminary work in [51] partially introduced the idea of leveraging markers mainly to extract semantic information from the environment to create meaningful 3D scene graphs, keeping the SLAM level largely independent from the markers. However, this work was constrained by its exclusive reliance on monocular cameras and exhibited limited scalability in large-scale environments with various illumination conditions, providing great potential for improvement. Additionally, since the baseline (UcoSLAM) mainly employs the markers detected in environments to enhance feature detection and tracking and apply loop closure constraints, our version also inherits the tight marker-to-keyframe bond and is not flexible enough to generate optimizable S-Graphs.

Therefore, this paper presents a more reliable vision-based approach combining a VSLAM system with a fiducial marker detection (both conventional markers as well as iMarkers), capable of generating a three-layered S-Graph of the environment using RGB-D cameras. It tightly couples the robot poses with structural semantic entities identified by fiducial markers, namely walls, doorways, rooms, and corridors, in a multi-layered, hierarchical, and optimizable S-Graph. Fig. 7.1 depicts an example of a generated three-layered graph for an indoor environment, interconnecting the robot poses with diverse structural and geometric priors about the scene. The principal contributions of this paper can be summarized as:

- A novel approach to create three-layered optimizable S-Graphs based on fiducial markers and supporting RGB-D visual sensors;
- A new solution for map reconstruction with a hierarchical representation procedure able to extract structural-level (*i.e.,* walls and doorways) and higher-level (*i.e.,* corridors and rooms) semantic entities;
- Utilizing the potential of semantic and geometric constraints imposed by fiducial markers for improving the quality of the reconstructed map and reducing localization errors;
- Revealing the concept and potential of iMarkers for robotic situational awareness applications.

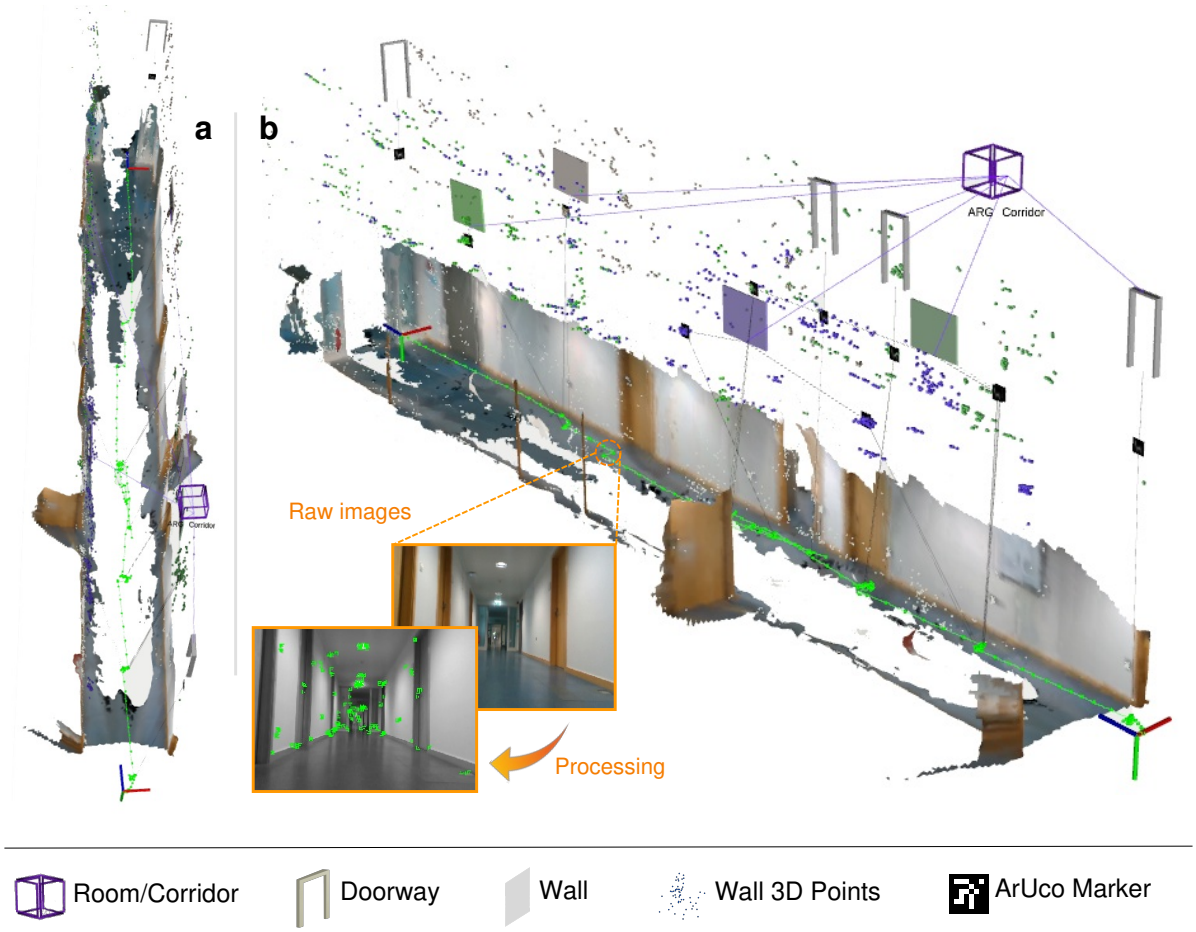


Figure 7.1: A reconstructed map with its hierarchical S-Graph representation generated by our framework, containing various detected structural-level entities and the connections among them: (**a**) the top view of the reconstructed map represented in 2D. (**b**) The final generated 3D view.

The rest of the paper is organized as follows: Section 7.3 explores previous efforts in the VSLAM domain, especially with respect to the use of structural-level data for improved reconstructed maps and the inclusion of fiducial markers. Section 7.4 details the proposed approach and its various modules. In Section 7.5, the evaluation criteria and an assessment of the effectiveness of the proposed method using real-world experimental data are provided. The functionality and performance of the proposed approach are discussed in detail in Section 7.6, and the paper finally concludes in Section 7.7.

7.3 Related Works

7.3.1 SLAM and 3D Scene Graphs

Over the years, VSLAM approaches have reached a higher level of maturity, and various innovative solutions have pushed the boundaries of this domain. Recent works in [248, 38] surveyed the state-of-the-art in VSLAM and discussed its current trends and possible directions. Accordingly, semantic VSLAM has evolved with methods such as [136, 137] estimating a geometric map and adding semantic objects in the environment for jointly optimizing the robot pose and semantic object landmarks. Sun *et al.* [120] proposed a deep learning-based method that extracts semantic information from the scene

and performs multi-object tracking based on it. However, the system does not work in real time and depends on a pre-processing step for segmentation. DS-SLAM [127] utilizes semantic information achieved by SegNet [217] for semantic mapping. The primary issue with DS-SLAM is the semantic segmentation limitations in covering various objects. PLPF-VSLAM [249] presents a VSLAM framework with an adaptive fusion of point-line-plane features, showing exciting results regardless of the richness of scene texture. Yang *et al.* [135] proposed another approach with dynamic object removal for generating static semantic maps. Although all of the above methods outperform their geometric VSLAM counterparts and can classify and map different semantic elements in the environment, they can still suffer from errors due to misidentification and the semantic elements' pose estimation errors. Thus, adding structural/topological constraints among various semantic elements could further increase the robustness of the environmental understanding.

To mitigate the limitations of semantic SLAM techniques, 3D scene graph approaches such as [159, 161, 160] generate hierarchical representations of the environment by interconnecting different semantic entities with suitable relations. While the above techniques consider SLAM and 3D scene graphs as two different optimization problems, recent methods like [157, 155] tightly couple SLAM graphs and 3D scene graphs for improving accuracy while generating meaningful, multi-layered hierarchical environment maps. Especially, S-Graphs [155] directly merges the SLAM and the scene graph into a common multi-layered and jointly optimizable factor graph. As an advantage, these S-Graphs not only offer a more comprehensive robotic situational awareness by combining geometric maps with diverse hierarchically organized semantic entities and their topological relationships within one graph, but they also lead to improved performance of localization and mapping on the SLAM level by exploiting semantic information. However, the generation of S-Graphs is so far limited to the exclusive use of LiDAR data.

7.3.2 Fiducial Markers and Marker-Based SLAM

While the aforementioned approaches for SLAM or semantic data extraction are based on natural visual features or objects of the environment, fiducial markers, which are intentionally added to the environment, can also serve as valuable tools for achieving enhanced scene understanding and mapping. Fiducial markers are used in robotics and Augmented Reality (AR) to encode information about objects on which they are applied, revealing the nature of each object and of the surrounding context to read-out devices. They can be classified into non-square (designed as circles [54, 56, 57], point sets [58, 59], or arbitrary visual patterns [63, 65, 71]), square (or matrix-based, such as ARToolkit [75], AprilTag [81], and ArUco [82]), and hybrid variations (such as DeepTag [84] and DynaTags [85]).

However, these conventional markers are visually invasive and optically distracting if they are placed in a normal environment. Furthermore, they are often printed on pure paper, which does not permit an application over a longer period of time without mechanical abrasion. Therefore, we recently introduced iMarkers, which are based on an innovative optical material called Cholesteric Spherical Reflectors (CSRs) [12]. CSRs are ~ 0.1mm diameter spheres of polymerized liquid crystal, exhibiting unique reflective properties. These make iMarkers invisible to humans, but not to adequately designed sensors, which can detect and read them even in visually complex and dynamic environments. Designed as flexible and flat foils that can be applied to hard or soft surfaces, iMarkers exhibit omnidirectional retro-reflectivity in a narrow wavelength band with circular polarization. This enables machine detection and readout from any direction, night or day, without

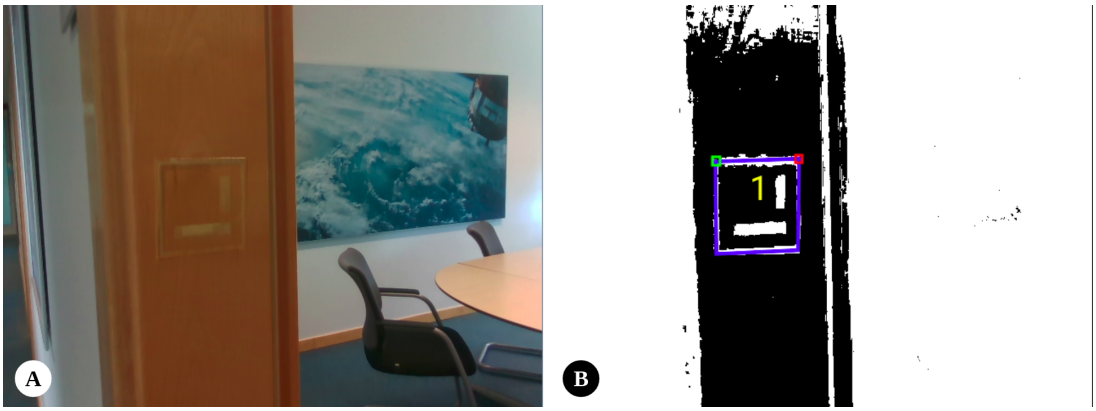


Figure 7.2: A transparent iMarker introduced by the authors: (**A**) an iMarker placed on a door frame captured by a normal camera, (**B**) the recognized iMarker with pose information, obtained by applying an ArUco detector on the image captured by the camera with a left-handed polarizer.

false positives [12]. The iMarkers can be designed to be transparent, as shown in Figure 7.2, but the CSR reflection band can also be tuned outside the visible spectrum, operating in the near- Infrared (IR) or Ultraviolet (UV) band, hence making the iMarkers undetectable by the human eye. Therefore, iMarkers would be non-intrusive and avoid attracting visual attention. While the iMarkers are totally novel from a material science perspective, the encoded geometric patterns we designed so far are the same as in the conventional ArUco markers; hence, the same ArUco detection and pose estimation algorithms can also be used for the iMarkers. Further details on the optical sensors and computer vision methods we developed for the detection and readout of the iMarkers can be found in [12].

Assuming that conventional fiducial markers are placed in the environment, the literature contains several approaches that explicitly use them in VSLAM. In this regard, UcoSLAM [150] is a marker-based VSLAM solution that utilizes visual features obtained from natural landmarks and ArUco markers. It works in keypoint-only, marker-only, and mixed modes, and is equipped with a marker-based loop closure detector, which requires placing distinct fiducial markers in the environment. TagSLAM [152] is another VSLAM approach that uses AprilTags to perform SLAM tasks. However, the system runs in marker-only mode and must constantly see the markers for localization and tracking stages. It should be noted that all of the mentioned approaches are focused on creating geometric maps and marker-based loop closure detection, and they do not employ the potential of fiducial markers for decoding semantic information. Thus, the preliminary work [51] of the authors of this paper introduced the idea of using ArUco markers for the designation of semantic entities in the environment, while a modified version of UcoSLAM (referred to as Semantic UcoSLAM) has then been leveraged for SLAM and the detection of the marked semantic entities. The primary drawbacks of this method were its low versatility (it is limited to monocular cameras), lack of scalability in larger environments, and tight marker-to-keyframe constraint inherited from the baseline, which avoids generating optimizable S-Graphs. Accordingly, the work introduced in this paper extends this preliminary approach by utilizing fiducial markers (including conventional as well as iMarkers based on the ArUco pattern) to generate a three-layered optimizable hierarchical S-Graph, incorporating in particular semantic objects with appropriate relational constraints, as detailed in the following.

7.4 Proposed Method

The presented framework is built upon ORB-SLAM 3.0 [108], and aims to provide more comprehensive reconstructed maps and their multi-level topological graph representations with semantic information derived from fiducial markers. The system takes advantage of a proper multi-threading architecture that successfully maintains the frame rate of the input and performs in real time. Moreover, it is implemented using Robot Operating System (ROS) to facilitate developing a modular design with simple integration, scalability, and communication capabilities. Since the first robotic applications of our proposed approach are related to buildings, *e.g.,* the autonomous monitoring of construction sites or the surveillance of indoor environments for security tasks, this paper restricts the set of included semantic entities to those that are the most important to describe the structure of building environments. Herein, walls and doorways are considered lower structural-level semantic entities that are annotated with a fiducial marker (ArUco-like conventional or iMarker), while rooms and corridors are considered higher-level semantic entities composed of the lower ones. However, the overall approach is, in principle, open to including any further semantic entities, which simply need to be annotated, in this case, with a fiducial marker and included from a mathematical point of view in the framework as described in the following.

All data are represented in the aforementioned S-Graph [155] format, *i.e.,* a common optimizable factor graph that merges the SLAM graph with the hierarchically ordered semantic information. Our framework computes structural-level elements' spatial positioning, leveraging fiducial markers affixed to them instead of relying on LiDAR-based odometry readings and planar surface extractions. It reconstructs a semantic map with hierarchical representations in the presence of ArUco-like markers and a database of high-level information about markers' affiliations with objects. The current framework version supports the RGB-D sensor and is extendable to monocular and stereo cameras, and leads to reconstructed map richness compared to the purely geometric maps generated by traditional VSLAM frameworks. In this paper, the focus is on the real-time generation of the S-Graph and the related algorithms, and less on the lower-level visual detection of the different included marker types, so we refer to [12] regarding the sensors and visual detection of iMarkers, if present in the scene.

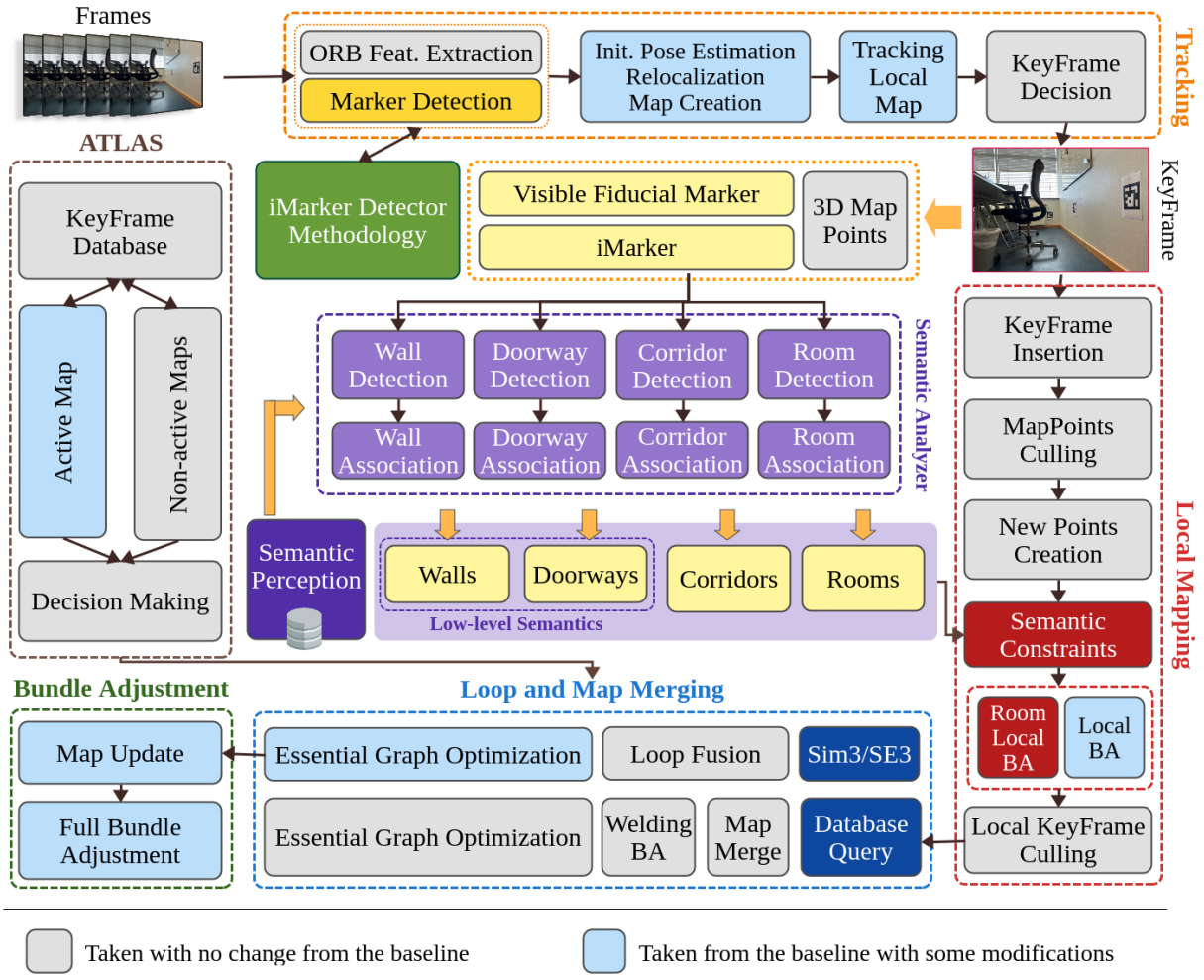


Figure 7.3: The primary system components and pipeline of the proposed approach. The components inherited from ORB-SLAM 3.0 with no changes are shown in gray, and the ones modified to match the current architecture are shown in light blue.

Figure 7.3 depicts the pipeline of the proposed methodology and its constituent components. The framework benefits from a multi-thread architecture for processing data, including *tracking*, *local mapping*, *loop and map merging*, and *marker detector*. The operation commences by processing the frames captured by an RGB-D camera and conveying them to the *tracking* module, where Oriented FAST and Rotated BRIEF (ORB) features and ArUco markers are extracted. The outcome of this module contains KeyFrame candidates with pose information, 3D map points, and possibly fiducial markers. If the *tracking* module decides that the current frame should be a KeyFrame, the *local mapping* module triggers to add the KeyFrame and points to the map, refining the map structure. Concurrently, the framework is being used to identify structural-level entities, including walls and doorways, and higher-level ones, containing corridors and rooms. The mentioned process takes place by leveraging pose information obtained from fiducial markers and a predefined *semantic perception* dictionary housing real-world data about the environment. Extracted semantic information is a resource for enhancing local Bundle Adjustment (BA) and KeyFrame culling the *local mapping*. It should be mentioned that the *Room local BA* module in *local mapping* is designed to connect the recently mapped walls and doorways within the current local KeyFrames to a newly detected room or corridor. Accordingly, it acts as a complementary module to the "local BA" to trigger the optimization process involving the lately mapped structural elements and incorporate them into a high-level

semantic entity, imposing new constraints. Thus, when added to the factor graph, the poses of these elements with respect to each other (such as the relative positions and orientations of walls and doorways within a room) provide valuable context that aids in further refining the map. The next impact is adjusting the poses of the KeyFrames and map points. The system constantly cooperates with an enhanced version of *Atlas* (the map manager module of ORB-SLAM 3.0) for establishing connections among disparate maps, representing the currently existing map (*i.e.,* active) and previously generated maps (*i.e.,* non-active). Loops and shared regions are detected within the active map and maps archived in *Atlas*, in the *loop and map merging* module. Finally, and in the case of loop detection and correction, the *global bundle adjustment* module is invoked to refine the constructed map further.

7.4.1 Fundamentals

The proposed method introduces four central coordinate systems at time t: the odometry frame of reference O, the camera coordinate system C_t, the marker coordinate system M_t, and the global coordinate system G_t. The vision sensor captures a set of frames $\mathbf{F} = \{\mathbf{f}\}$, with each frame $\mathbf{f} = \{t, \mathbf{T}, \delta\}$ containing the camera pose $\mathbf{T} \in SE(3)$ acquired through the transformation of C_t to G_t, as well as intrinsic camera parameters δ. Each frame f undergoes sub-sampling into an image pyramid and is processed by ORB feature extractor to obtain a set of key points for selecting KeyFrames, denoted as $\mathbf{K} = \{k\} \subset \mathbf{F}$. These KeyFrames contain feature points $\mathbf{P} = \{\mathbf{p}\}$ and fiducial markers $\mathbf{M} = \{\mathbf{m}\}$ (regardless of their type) used for detecting structural-level and semantic entities. A feature point $\mathbf{p} = \{\mathbf{x}, \mathbf{v}, \hat{\mathbf{d}}\}$ is characterized by its corresponding 3D position $\mathbf{x} \in \mathbb{R}^3$, viewing direction $\mathbf{v} \in \mathbb{R}^3$, and descriptor $\hat{\mathbf{d}}$. On the other hand, each marker $\mathbf{m} = \{id, t, s, \mathbf{p}\}$ holds unique ArUco marker identifier $m_i \in \mathbb{N}$, length $s \in \mathbb{R}$, and pose $\mathbf{p} \in SE(3)$ derived from the transformation of M_t to G_t. Consequently, the final representation of the reconstructed map of the environment $\mathbf{E}$ is defined as follows:

$$\mathbf{E} = \{\mathbf{K}, \mathbf{P}, \mathbf{M}, \mathbf{W}, \mathbf{D}, \mathbf{R}\} \tag{7.1}$$

where $\mathbf{W} = \{\mathbf{w}\}$ encompasses the detected walls within the environment, in which each wall $\mathbf{w} = \{t, \mathbf{q}, \mathbf{m_w}\}$ holds the wall equation $\mathbf{q} \in \mathbb{R}^4$ and the list of attached markers $\mathbf{m_w} \subset \mathbf{M}$. $\mathbf{D} = \{\mathbf{d}\}$ comprises the doorways found in the environment, where each doorway $\mathbf{d} = \{t, m_d, \mathbf{p}\}$ contains attached marker $\mathbf{m_d} \in \mathbf{M}$ and pose $\mathbf{p} \in SE(3)$ computed from M_t to G_t transformation. Finally, $\mathbf{R} = \{\mathbf{r}\}$ represents the set of rooms/corridors present in the environment, each $\mathbf{r} = \{t, \mathbf{r_c}, \mathbf{r_w}\}$ with a center point $\mathbf{r_c} \in \mathbb{R}^3$ and a list of walls $\mathbf{r_w} \subset \mathbf{W} = (w_1 ... w_n) | w_i \in \mathbb{N}$ that constitute the boundaries of the room or corridor.

7.4.2 Structural-Level and Higher-Level Semantic Entities

Reconstructing a rich semantic map of the environment incorporating the mentioned entities entails employing diverse methodologies, which will be discussed in this section. In the proposed approach, fiducial markers (which could be conventional or iMarkers) thus play a vital role, and their synergy with the system, in conjunction with the *semantic perception* module, aids in recognizing the entities of interest. Accordingly, the proposed framework takes advantage of fiducial markers, not only to add geometric constraints for map creation but also to enrich them with desired building-related semantic entities acquired with the aid of markers. It is worth emphasizing that the *semantic perception* dictionary encodes exclusively the *marker-ids* corresponding to rooms and doorways, obviating the need for any supplementary pose information of the labeled objects to be incorporated.

Markers. Fiducial markers are crucial reference points in our framework, enabling the system to interpret and contextualize semantic data in the environment. It has to be mentioned that the fiducial markers contain no direct information about the characteristics of the labeled items, such as pose in the global reference and width/height. Owing to their distinctive textures and the capacity to compute pose information ($\mathbf{p} \in SE(3)$), fiducial markers are primary sources of information in the proposed work for identifying and labeling targeted semantic entities, *i.e.,* walls and doorways. Each marker $\boldsymbol{m}_i$ in G_t is constrained by the KeyFrame K_i observing it, which can be formulated as:

$$c_{m_i}({}^{G}\boldsymbol{K}_i, {}^{G}\boldsymbol{m}_i) = \| {}^{L}\boldsymbol{m}_i \boxplus {}^{G}\boldsymbol{K}_i \boxminus {}^{G}\boldsymbol{m}_{i_1} \|^2_{\boldsymbol{\Lambda}_{\tilde{\boldsymbol{m}}_i}} \tag{7.2}$$

where ${}^{L}\boldsymbol{m}_i$ represents the locally observed fiducial marker's pose, $\boxplus$ and $\boxminus$ refer to the composition and inverse composition, $\| \dots \|$ is the Mahalanobis distance, and $\boldsymbol{\Lambda}_{\tilde{\boldsymbol{m}}_i}$ is information matrix associated with $\tilde{m}_i$.

Walls. The procedure employed to identify wall surfaces, which serve as the planar substrates on which ArUco markers and feature points are situated, relies on pose information derived from detected markers. Wall detection occurs whenever a fiducial marker is visited within the current KeyFrame. Hence, by accessing the real-world environment data obtained from the *semantic perception* module, in cases where the marker's identifier is not found within the list of markers associated with doorways, the planar equation of the wall is calculated using the poses of the attached marker and the surrounding map points. It should be noted that this work assumes that all fiducial markers are affixed directly to the walls in an environment. Consequently, the equations characterizing these walls are obtained based on the poses of the markers attached.

Each wall $\boldsymbol{w}_i$ in G_t is represented by ${}^{G}\boldsymbol{w}_i = \begin{bmatrix} {}^{G}\boldsymbol{n}_i & {}^{G}d \end{bmatrix}$, where ${}^{G}\boldsymbol{n}_i = \begin{bmatrix} n_x & n_y & n_z \end{bmatrix}^T$ denotes the normal vector of the wall and d represents the distance of the wall $\boldsymbol{w}_i$ from the origin in the global coordinate system. The vertex node of the wall within the graph is denoted as ${}^{G}\boldsymbol{w_i} = [{}^{G}\phi, {}^{G}\theta, {}^{G}d]$, where ${}^{G}\phi$ and ${}^{G}\theta$ represent the azimuth and elevation angles of the wall in the global coordinate G_t, respectively. Consequently, the cost function associated with each marker ${}^{G}\boldsymbol{m}_i$ affixed to the wall ${}^{G}\boldsymbol{w}_i$ can be calculated as follows:

$$c_{w_i}({}^{G}\boldsymbol{w}_i, {}^{G}\boldsymbol{m}_i) = \| [{}^{M}\delta\phi_{w_{i m_i}}, {}^{M}\delta\theta_{w_{i m_i}}, {}^{M}d_{w_i}]^T \|^2_{\boldsymbol{\Lambda}_{\tilde{\boldsymbol{w}}_i}} \tag{7.3}$$

where ${}^{M}\delta\phi_{w_{i m_i}}$ represents the disparity between the azimuth angle of wall w_i and its attached marker m_i in M_t, ${}^{M}\delta\theta_{w_{i m_i}}$ denotes the difference in elevation angles between the two. In contrast, ${}^{M}d_{w_i}$ signifies the perpendicular distance separating the wall from the marker. This distance should ideally be zero for given marker–wall pairings.

Corridors (Two-Wall Rooms). Our framework leverages an adapted version of the "room segmentation" methodology originally presented in *S-Graphs*, wherein markers are employed for detecting walls belonging to rooms. In this context, a corridor is defined as a room in the environment in which two parallel walls are labeled with ArUco markers. Due to the complexities associated with detecting rooms with diverse layouts, this work extends its definition to include rooms with inaccessible walls as corridors.

A corridor ${}^{G}\mathbf{r}_x = [{}^{G}\mathbf{w}_{x_{a_1}}, {}^{G}\mathbf{w}_{x_{b_1}}]$ encompasses wall planes aligned with the x-axis. To calculate the center point of a corridor ${}^{G}\mathbf{r}_{x_i}$, the two equations representing the x-wall planes are employed in conjunction with the center point ${}^{G}\mathbf{c}_i$ of the marker $\mathbf{m}_i$ in the following manner:

$$ {}^{G}\mathbf{k}_{x_i} = \frac{1}{2}|{}^{G}d_{x_{a_1}}| \cdot {}^{G}\mathbf{n}_{x_{a_1}} - |{}^{G}d_{x_{b_1}}| \cdot {}^{G}\mathbf{n}_{x_{b_1}} + |{}^{G}d_{x_{b_1}}| \cdot {}^{G}\mathbf{n}_{x_{b_1}} \tag{7.4} $$

$$ {}^{G}\boldsymbol{\eta}_{x_i} = {}^{G}\hat{\mathbf{k}}_{x_i} + {}^{G}\mathbf{c}_i - [\ {}^{G}\mathbf{c}_i \cdot {}^{G}\hat{\mathbf{k}}_{x_i}]\ \cdot {}^{G}\hat{\mathbf{k}}_{x_i} \tag{7.5} $$

where ${}^{G}\boldsymbol{\eta}_{x_i}$ represents the center point of the corridor ${}^{G}\mathbf{r}_{x_i}$ and ${}^{G}\hat{\mathbf{k}}_{x_i}$ is derived from ${}^{G}\hat{\mathbf{k}}_{x_i} = {}^{G}\mathbf{k}_{x_i}/\|{}^{G}\mathbf{k}_{x_i}\|$. The center point ${}^{G}\mathbf{c}_i$ of the marker is determined based on the marker's pose within the G frame. Notably, the computation of the center for a two-wall room in the y direction follows a similar procedure. The cost function to minimize the corridor's vertex node and its corresponding wall planes is defined as follows:

$$ c_{\boldsymbol{r}_{x_i}}({}^{G}\boldsymbol{r}_{x_i}, [{}^{G}\mathbf{w}_{x_{a_1}}, {}^{G}\mathbf{w}_{x_{b_1}}, {}^{G}\mathbf{c}_i]) = \sum_{t=1,i=1}^{T,K} \|{}^{G}\hat{\boldsymbol{\eta}}_{x_i} - f({}^{G}\tilde{\mathbf{w}}_{x_{a_1}}, {}^{G}\tilde{\mathbf{w}}_{x_{b_1}}, {}^{G}\mathbf{c}_i)\|^2_{\boldsymbol{\Lambda}_{\tilde{\boldsymbol{r}}_{i,t}}} \tag{7.6} $$

where $f({}^{G}\tilde{\mathbf{w}}_{x_{a_1}}, {}^{G}\tilde{\mathbf{w}}_{x_{b_1}}, {}^{G}\mathbf{c}_i)$ is a mapping function that associates the wall planes with the corridor's center point.

Rectangular Rooms (Four-Wall Rooms). In the scenario where a room in the environment consists of four walls, each labeled with ArUco markers (*i.e.,* two pairs of perpendicular labeled walls), the room is represented as ${}^{G}\mathbf{r}_i = [{}^{G}\mathbf{w}_{x_{a_1}}\ {}^{G}\mathbf{w}_{x_{b_1}}\ {}^{G}\mathbf{w}_{y_{a_1}}\ {}^{G}\mathbf{w}_{y_{b_1}}]$. The center point ${}^{G}\boldsymbol{\rho}_i$ of the room ${}^{G}\mathbf{r}_i$ is computed using the following equation:

$$ {}^{G}\mathbf{q}_{x_i} = \frac{1}{2}\big[|{}^{G}d_{x_{a_1}}| \cdot {}^{G}\mathbf{n}_{x_{a_1}} - |{}^{G}d_{x_{b_1}}| \cdot {}^{G}\mathbf{n}_{x_{b_1}}\big] + |{}^{G}d_{x_{b_1}}| \cdot {}^{G}\mathbf{n}_{x_{b_1}} \tag{7.7} $$

$$ {}^{G}\mathbf{q}_{y_i} = \frac{1}{2}\big[|{}^{G}d_{y_{a_1}}| \cdot {}^{G}\mathbf{n}_{y_{a_1}} - |{}^{G}d_{y_{b_1}}| \cdot {}^{G}\mathbf{n}_{y_{b_1}}\big] + |{}^{G}d_{y_{b_1}}| \cdot {}^{G}\mathbf{n}_{y_{b_1}} \tag{7.8} $$

$$ {}^{G}\boldsymbol{\rho}_i = {}^{G}\boldsymbol{q}_{x_i} + {}^{G}\boldsymbol{q}_{y_i} \tag{7.9} $$

where the equation is positive if $|d_{x_1}| > |d_{x_2}|$. The cost function to minimize the room's vertex node and its corresponding wall planes is defined as follows:

$$ c_{\boldsymbol{\rho}}({}^{G}\boldsymbol{\rho}, [{}^{G}\mathbf{w}_{x_{a_i}}, {}^{G}\mathbf{w}_{x_{b_i}}, {}^{G}\mathbf{w}_{y_{a_i}}, {}^{G}\mathbf{w}_{y_{b_i}}]) = \sum_{t=1,i=1}^{T,S} \|{}^{G}\hat{\boldsymbol{\rho}}_i - f({}^{G}\tilde{\mathbf{w}}_{x_{a_i}}, {}^{G}\tilde{\mathbf{w}}_{x_{b_i}}, {}^{G}\tilde{\mathbf{w}}_{y_{a_i}}, {}^{G}\tilde{\mathbf{w}}_{y_{b_i}})\|^2_{\boldsymbol{\Lambda}_{\tilde{\boldsymbol{\rho}}_{i,t}}} \tag{7.10} $$

where $f({}^{G}\tilde{\mathbf{w}}_{x_{a_i}}, {}^{G}\tilde{\mathbf{w}}_{x_{b_i}}, {}^{G}\tilde{\mathbf{w}}_{y_{a_i}}, {}^{G}\tilde{\mathbf{w}}_{y_{b_i}})$ is a mapping function that associates wall planes with the room's center point.

Doorways. In this case, the pose information of ArUco markers placed on a door frame is employed to define the doorway in the map. The procedure involves visiting fiducial markers present in the current KeyFrame and verifying their association with doorways through the utilization of the *semantic perception* module. Once confirmed, the pose of the visited marker is designated for the doorway.

Accordingly, the cost function for each doorway $\boldsymbol{d}_i$ in G_t and its corresponding room (or corridor) ${}^{G}\mathbf{r}_i$ is computed as follows:

$$ c_{d_i}({}^{G}\boldsymbol{d}_i, {}^{G}\boldsymbol{r}_i) = \|{}^{G}\hat{\boldsymbol{\delta}}_{d_i,r_j} - f({}^{G}\boldsymbol{d}_i, {}^{G}\boldsymbol{r}_i)\|^2_{\boldsymbol{\Lambda}_{\tilde{\boldsymbol{d}}_{i,t}}} \tag{7.11} $$

where ${}^{G}\hat{\boldsymbol{\delta}}_{d_i,r_j}$ represents the relative distance between the door and the room and $f({}^{G}\boldsymbol{d}_i, {}^{G}\boldsymbol{r}_j)$ maps the relative distance among their nodes.

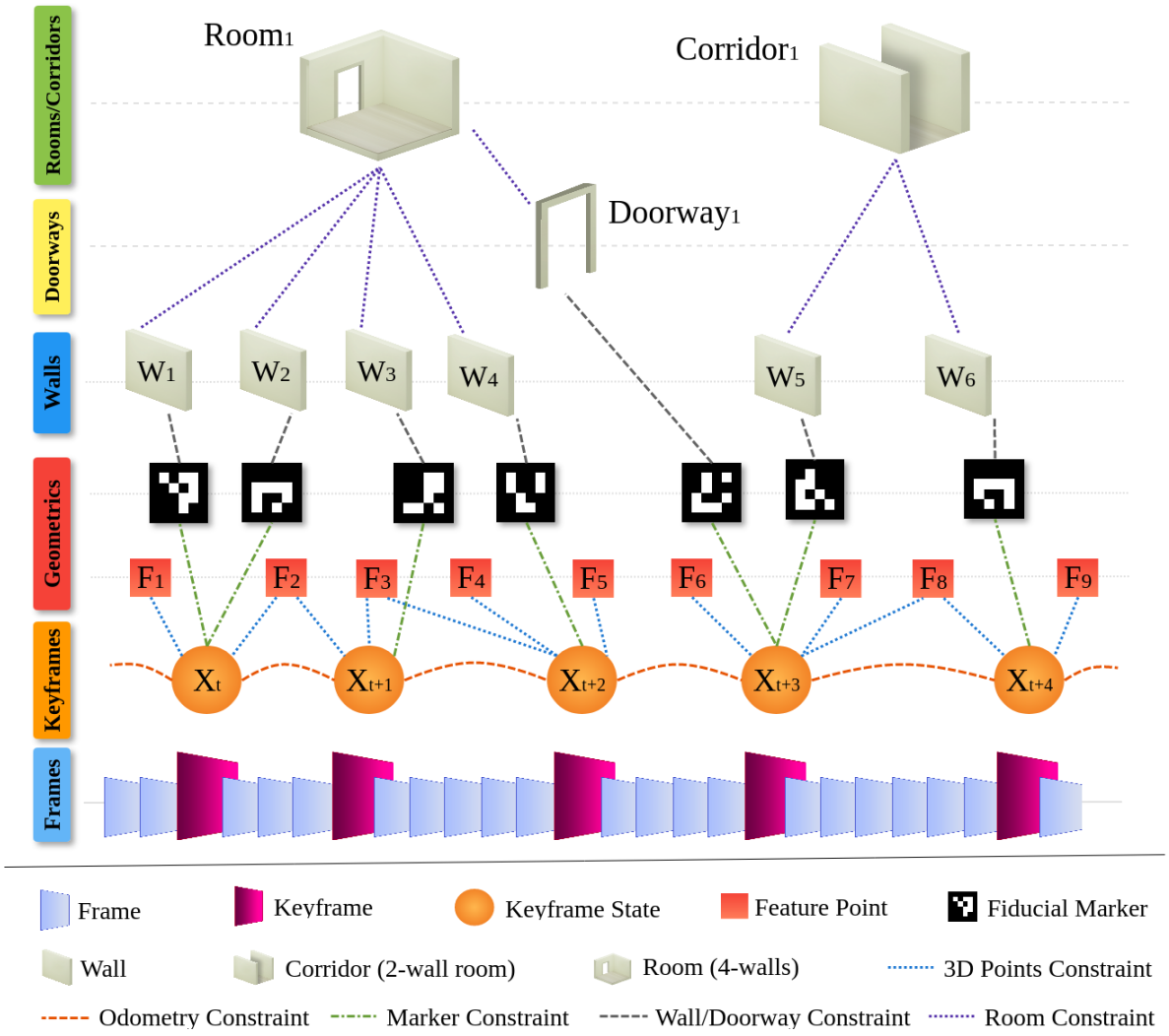


Figure 7.4: The hierarchical S-Graph representation of our proposed work incorporates semantic constraints, including walls, doorways, corridors, and rooms. The pre-existing geometric constraints have been complemented with semantic constraints acquired by fiducial markers.

7.4.3 The Final S-Graph

The structure of the final S-Graph generated by our framework is depicted in Fig. 7.4. Accordingly, the primary tracking information sources are located in the KeyFrames, which contain geometric data and pose information of objects, including 3D points and visited ArUco markers. The constraints among the mentioned objects guarantee proper computation of the odometry and loop closure detection. Structural-level entities, including walls and doorways, are linked to constraints associated with the mapped fiducial markers and establish next-level constraints with higher-level entities, including rooms and corridors.

7.4.4 Inclusion of iMarkers

As conventional fiducial markers may face problems with visual clutter, they can be seamlessly replaced with iMarkers in our framework. Since the production of our iMarkers is still in an experimental stage, larger amounts of iMarkers could not yet be produced and included in the assessment. However, we applied a combination of conventional printed

Table 7.1: The characteristics of the collected indoor dataset.

Seq. *	Duration	Description
Seq-01	06min 27s	Two rooms connected via a door
Seq-02	07min 55s	A corridor connected to a room and another corridor
Seq-03	12min 32s	Five rooms connected to a corridor
Seq-04	07min 34s	Two corridors connected via a landing area
Seq-05	16min 42s	Four corridors connected to a room, forming a loop
Seq-06	01min 44s	A single room connected to a corridor

* Data were stored as packages of *rosbag* files.

and a few transparent iMarkers, all with an ArUco pattern, for these experiments here. The project's final future vision is to label desired semantic entities with invisible iMarkers only instead of using printed ones. The performance and usage of the mentioned markers in robotics will be evaluated in detail in Section 7.5.

7.5 Evaluation

To evaluate the performance and robustness of the proposed method compared to other existing frameworks, various experiments have been performed using the proposed approach, UcoSLAM [150], as a well-known marker-based method, our previous methodology known as Semantic UcoSLAM [51], and ORB-SLAM 3.0 [108] as the baseline and a reliable VSLAM methodology. It should be noted that since the proposed approach is more *marker-based* than a fully *semantic* VSLAM, evaluating against existing semantic VSLAM methodologies can be unfair. The reason lies in the fundamental dissimilarities in the context, where *semantic* VSLAM requires more computational resources (mainly GPU-dependent computer vision methods), and is designed to extract a broader range of semantic information. Additionally, and due to less noisy outputs of LiDAR sensors compared to vision-based sensors, *S-Graphs+* [156], as a LiDAR-based framework, has been used to produce ground truth data. A computer equipped with an *11th Gen. Intel Core-i9 @2.60GHz* processor and 32 GigaBytes of memory was used for the evaluation mentioned.

7.5.1 Evaluation Setup

For evaluating the performance of the proposed method in real-world conditions, a 3D LiDAR sensor and an *Intel RealSense Depth Camera D435i* were mounted on legged robots, including *Boston Dynamics Spot®* and *Unitree Go1* (shown in Figure 7.5). The mounted camera captures the scene viewed by the robot at a rate of 25 Frames Per Second (FPS). The robots collected data from sensors as they traversed various indoor environments with different room and corridor configurations. Each wall and door of the rooms and corridors was labeled with 8cm × 8cm ArUco-like markers, and the unique identifiers of the markers were stored in a database to feed the proposed method (*i.e.,* the *semantic perception* module in Fig. 7.3). As previously mentioned, the environment for data collection has been prepared using a combination of printed ArUco markers and a few transparent iMarkers. Accordingly, the characteristics of the collected datasets are presented in Table 7.1.



Figure 7.5: Dataset collected to evaluate the proposed method: (**A**) the legged robots used for data collection, (**B**) some instances of the environment prepared for data collection.

7.5.2 Experimental Results

Accuracy. To validate the accuracy of the proposed method in comparison to its baseline and ground truth, ATE measurements have been employed in this paper. Regarding the evaluation results presented in Table 7.2, it becomes evident that our approach outperforms its baseline counterpart and other frameworks across various scenarios. This enhancement can be attributed to the proposed method's ability to introduce new constraints to the map by associating structural-level and semantic entities. It is noteworthy that the improvements are particularly pronounced in comparison to the two marker-based methodologies. While on *Seq-02* and *Seq-05*, ORB-SLAM 3.0 exhibits slightly better performance than our method, the difference is negligible (*i.e.,* $<$ 3cm). This discrepancy could be due to the noisy detection of the fiducial markers as the primary source of semantic information in real-world scenarios with changing light conditions. However, it is important to emphasize that the proposed approach, in addition to improving ATE in most cases, can generate a three-layered situational graph of the environment. Accordingly, Figure 7.6 depicts some qualitative results alongside the accuracy of the proposed approach against the LiDAR-based benchmark. The qualitative results in sub-figures *A* and *C* show that the proposed approach can reconstruct rich maps of the environments by mapping the detected structural-level semantic objects with the aid of fiducial markers and calculating the presence of high-level semantic entities, including rooms and corridors. It also represents the connection among the mentioned entities in the form of the hierarchical scene graphs, described in Figure 7.4.

The performance of the iMarkers. To verify the applicability and potential of our described iMarkers for the first time in robotics, the door frames in *Seq-06* were labeled with a prototype of transparent iMarkers. The reconstructed map using the iMarkers is depicted in Figure 7.6. Given that the sole distinction between iMarkers and conventional printed ArUco markers lies in the detection step, the resulting semantic map remained unaltered. The authors believe this will provide proper room for future investigation on utilizing such iMarkers in their future works.

Execution time and computational cost. As mentioned in Section 7.5.1, data instances used for evaluation were recorded at 25fps. Validation results on the dataset showed that the designed system, thanks to its multi-threaded architecture shown in Fig-

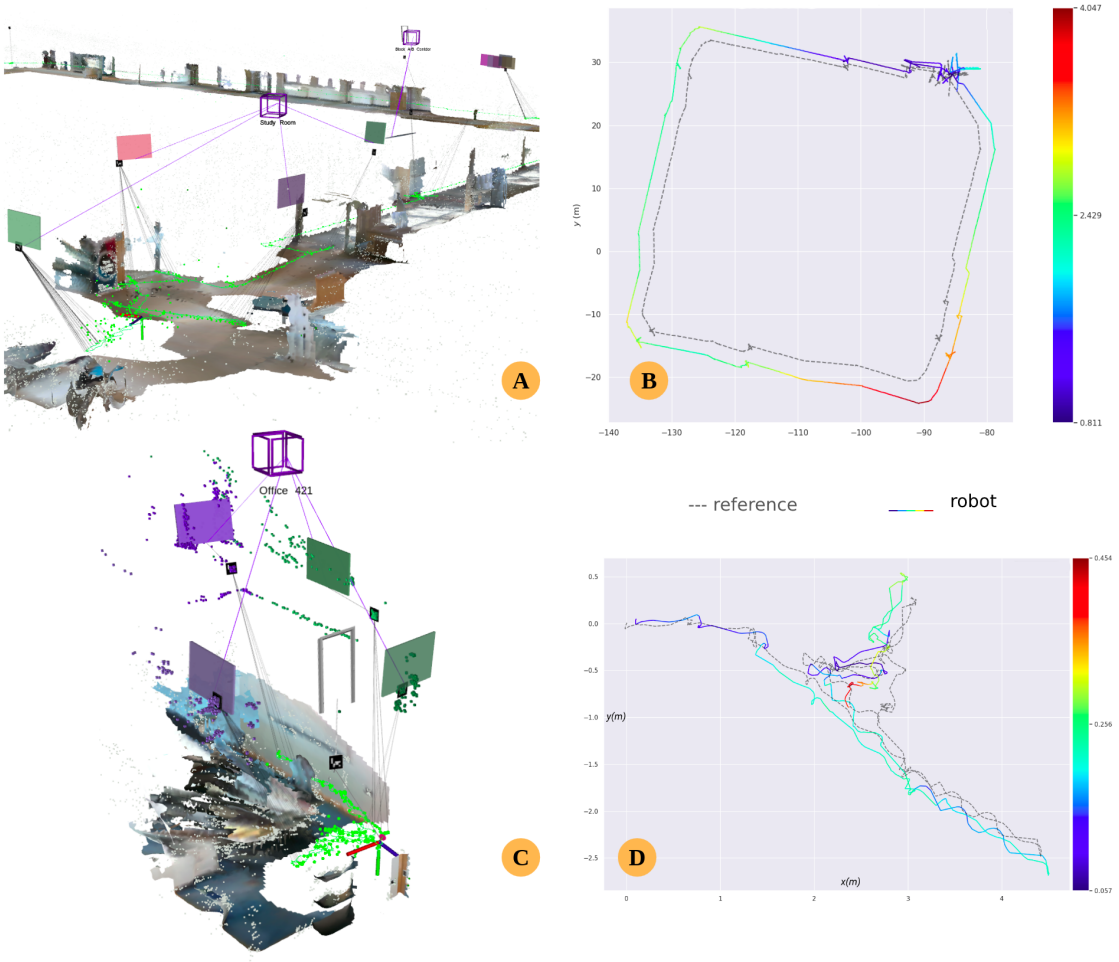


Figure 7.6: The qualitative results and Absolute Trajectory Error (ATE) of the proposed approach *w.r.t.* translation in *meters* on *Seq-05* (**A**,**B**) and *Seq-06* (**C**,**D**). The dotted lines in the charts are LiDAR ground truth reference values, and the solid lines show the trajectory of the robot. In an ideal representation, the shapes should perfectly match, while the calculated pose errors (from low errors shown in navy blue to high errors shown in dark red) in the Visual SLAM system are inevitable.

ure 7.3, could efficiently handle the computational load and continue performing in real-time at 25fps. It should be noted that the computational hurdle is not evenly distributed across all the framework modules. For instance, extracting ORB features in the *tracking* module is performed in real-time, while in parallel, an ArUco marker detector seeks markers in each frame and integrates the detected marker poses into the framework via ROS-based communication. As a module with intensive calculations, *local mapping* and its underlying optimization processes are triggered periodically based on new KeyFrame detections. However, it operates asynchronously with respect to the *tracking* module and without directly influencing the frame rate. Identifying structural- and higher-level entities and querying the *semantic perception* dictionary are also not computationally intensive, and require 10%–15% of the processing time. Finally, the most computationally demanding modules are *loop and map merging* and *global bundle adjustment*, which operate less frequently and asynchronously, triggered only when significant map updates are necessary.

Moreover, in terms of the required processing time, experiments showed that all the

Table 7.2: Evaluation results on the collected dataset using Root Mean Square Deviation (RMSE) error in *meters* and Standard Deviation (STD). The best results are boldfaced and the second best are underlined. Our method outperforms the state-of-the-art in most of the sequences.

	RMSE					
	Seq-01	**Seq-02**	**Seq-03**	**Seq-04**	**Seq-05**	**Seq-06**
Proposed	**0.5127**	<u>0.6662</u>	**2.3555**	**0.4479**	<u>2.1794</u>	**0.2189**
UcoSLAM [150]	5.7996	3.0521	3.3034	2.1573	15.0184	1.5601
ORB-SLAM 3.0 [108]	<u>0.5351</u>	**0.6484**	<u>2.5011</u>	<u>0.4895</u>	**2.1404**	<u>0.2479</u>
Sem. UcoSLAM [51]	4.9437	2.8363	2.5154	1.9154	4.6672	1.5552
	STD					
	Seq-01	**Seq-02**	**Seq-03**	**Seq-04**	**Seq-05**	**Seq-06**
Proposed	**0.2454**	**0.3332**	**0.7441**	**0.2422**	**0.7107**	**0.0796**
UcoSLAM [150]	3.1814	1.3999	1.2332	1.2284	6.1595	0.8055
ORB-SLAM 3.0 [108]	<u>0.2572</u>	<u>0.3334</u>	0.8602	<u>0.2653</u>	<u>0.7366</u>	<u>0.0815</u>
Sem. UcoSLAM [51]	2.7065	1.3191	<u>0.8582</u>	1.1547	2.3891	0.8014

evaluated frameworks in Table 7.2 work in real-time (and almost in real-time) on the utilized computer. In this regard, ORB-SLAM 3.0 functions faster than others, in the range of 33 to 38 milliseconds per frame (ms/frame). The proposed framework requires more time (an average of 46ms/frame) due to adding new constraints and calculations, but it still guarantees low latency rates during processing. Similar outcomes are obtained for UcoSLAM and its semantic version, with the difference that the baseline is almost 1.4× faster due to less processing overhead. The point to be emphasized here is the trade-off between map richness (and, therefore, supplying improved situational awareness capabilities) and processing time in different applications where a slightly higher processing load is worthwhile to acquire improved functionality.

7.6 Discussions

Regarding the evaluation results and comparing them with other marker-based state-of-the-art works, the authors would like to highlight the innovative nature of the methodology presented in this paper. The proposed approach employs fiducial markers to enhance map reconstruction and scene representation in a visual SLAM framework by avoiding the significant computational overhead of integrating computer vision and deep learning-based scene understanding modules. By concentrating only on scene elements augmented with fiducial markers, including "quasisemantic structural-level objects" (*i.e.,* walls and doorways) and "calculable higher-level semantic entities" (including corridors and rooms), the framework simplifies the interpretation process to paramount components. This selective scene comprehension enables better digital twin creation and enriched scene graph generation, ultimately producing more informative maps for robotic tasks where situational awareness is crucial. For instance, in a task like navigation or path planning, the proposed framework generates maps with detailed environmental information, including the presence (and, if so, the locations) of walls, doorways, and corridors. This offers a substantial advantage over purely geometric maps created using classic methods like ORB-SLAM 3.0 [108] or UcoSLAM [150], which may require more semantic depth for a robot performing path planning.

On the other hand, despite not utilizing extensive computer vision frameworks for

semantic scene segmentation or object detection/tracking, the introduced method derives the required semantic information from marker poses and unique identifiers to enhance map reconstruction. Accordingly, it allows rapid mapping of objects and the imposition of additional constraints, leading to higher levels of richness and usability of the reconstructed maps without heavy computational demands. Needless to say, the approach is designed to map *any labeled static object* for rapid and cost-effective map enrichment, supporting more robust and situationally aware robotic applications.

Another important note is that the proposed approach is dependent on the presence of markers, which could pose limitations in environments where marker placement/detection is challenging. However, the proposed solution offers a deliberate trade-off between "simplicity" and "performance" while guaranteeing richer reconstruction maps with higher estimated robot poses. Although the demand for detectable markers may restrict application in specific scenarios, the method offers many benefits in environments where markers can be appropriately placed.

Considering the significance of the framework's scalability in working fine in larger environments and the simplicity of covering more semantic entities (items labeled with fiducial markers) to generate richer maps, the proposed framework is built using ROS, which inherently supports scalability through its modular architecture. It also provides distributed processing so the designed nodes (modules) handle diverse tasks and maintain parallel execution. Accordingly, the proposed method scales efficiently with increasing the environment's scope and complexity. Observations while running the experiments imply that the framework's multi-threaded architecture and fine-tuned workload division can lead the proposed approach to maintain its real-time performance.

Finally, the authors would like to underscore that the proposed approach is a marker-based VSLAM solution inspired by *S-Graphs* [155]. While the LiDAR-based version is more accurate, comparing it directly to the proposed VSLAM version is unjust due to the fundamental differences between the sensing modalities. On the other hand, the proposed framework is cost-effective and captures textures and rich visual features, simplifying the process of "understanding" the environment in which the robot is functioning (especially with the aid of markers). However, reconstructing the situational graph and presenting it in a 3D hierarchy similar to the ones provided by the *S-Graphs* methodology requires mathematical operations inherited from *S-Graphs* and adapted to the proposed framework.

7.7 Conclusions

This paper introduced a VSLAM framework that effectively harnesses the outputs provided by RGB-D cameras to achieve highly accurate map reconstruction in the generation of S-Graphs. The proposed approach employs pose and topological information derived from strategically positioned ArUco markers, both conventional printed markers and our invented iMarkers, within indoor environments to detect semantic objects, including walls, doorways, corridors, and rooms. It utilizes the added semantic entities and the topological constraints among them for an elevated reconstructed map quality. Considering the evaluations performed on a real-world dataset collected by legged robots and benchmarked against a LiDAR-based framework as the ground truth, the proposed method showed an accuracy and performance improvement compared to state-of-the-art works.

As the proposed framework is part of a broader research project, in future works, the authors intend to determine transparent objects (*e.g.,* windows, mirrors, and glass doors) using the iMarkers due to their challenging recognition using computer vision algo-

rithms and potential difficulties for robots performing SLAM tasks. Moreover, supporting more visual (*i.e.,* mono and stereo cameras) and inertial (*i.e.,* Inertial Measurement Unit (IMU)) sensors along with the efficient implementation of modules is another target of future work. The authors also plan another extension of their work to reserve fiducial markers for higher-level tasks (*i.e.,* incorporating specific semantic information of rooms and corridors) instead of utilizing them for structural-level semantic object detection, which can be replaced by scene semantic segmentation.

8

Visual S-Graphs (vS-Graphs)

The fifth paper [168] was submitted to **IEEE Robotics & Automation Letters (RA'L)** in March 2025 and is currently under review[1]. This work represents one of the key contributions of my Ph.D., proposing a Visual SLAM framework that integrates vision-based scene understanding with map reconstruction and interpretable graph-based representation. The system can generate a comprehensible digital twin of the operational environment. While the use of markers or *iMarkers* is optional, their inclusion further enhances semantic labeling and interpretation.

Paper Card

Title: vS-Graphs: Integrating Visual SLAM and Situational Graphs through Multi-level Scene Understanding
Authors: A. Tourani, S. Ejaz, H. Bavle, M. Fernandez-Cortizas, D. Morilla-Cabello, J.L. Sanchez-Lopez, and H. Voos
Venue: IEEE Robotics & Automation Letters (under review)
DOI: `https://doi.org/10.48550/arXiv.2503.01783`
Video: `https://youtu.be/5kbgUucvQos?si=mjsj58rbKCyxXGLj`
Web-page: `https://snt-arg.github.io/vsgraphs-results/`
Repository: `https://github.com/snt-arg/visual_sgraphs`

Copyright

© 2026 by the authors. This article is an open-access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (`https://creativecommons.org/licenses/by/4.0/.`)

[1] A revised version was submitted in November 2025.

8.1 Abstract

Current Visual Simultaneous Localization and Mapping (VSLAM) systems often struggle to create maps that are both semantically rich and easily interpretable. While incorporating semantic scene knowledge aids in building richer maps with contextual associations among mapped objects, representing them in structured formats like scene graphs has not been widely addressed, encountering complex map comprehension and limited scalability. This paper introduces vS-Graphs, a novel real-time VSLAM framework that integrates vision-based scene understanding with map reconstruction and comprehensible graph-based representation. The framework infers structural elements (*i.e.,* rooms and floors) from detected building components (*i.e.,* walls and ground surfaces) and incorporates them into optimizable 3D scene graphs. This solution enhances the reconstructed map's semantic richness, comprehensibility, and localization accuracy. Extensive experiments on standard benchmarks and real-world datasets demonstrate that vS-Graphs achieves an average of **15.22**% accuracy gain across all tested datasets compared to state-of-the-art VSLAM methods. Furthermore, the proposed framework achieves environment-driven semantic entity detection accuracy comparable to that of precise LiDAR-based frameworks, using only visual features.

The code is publicly available at `https://github.com/snt-arg/visual_sgraphs` and is actively being improved. Additionally, a web page with more media and evaluation results is available at `https://snt-arg.github.io/vsgraphs-results/`.

8.2 Introduction

Robust environment understanding, a core foundation of robots' situational awareness [163] in the context of Simultaneous Localization and Mapping (SLAM), relies heavily on sensor quality and modality. While diverse sensors, *e.g.,* Light Detection And Ranging (LiDAR) and cameras, have been employed in SLAM, vision sensors offer a cost-effective solution for rich map reconstruction, forming a distinct category titled VSLAM [250]. Among vision sensors, RGB-D cameras provide complementary visual and depth cues, overcoming monocular limitations by producing dense point clouds that capture fine spatial details, enabling precise detection, localization, and mapping of environmental elements [248]. To further enhance Visual SLAM (VSLAM) performance, visual perception techniques are integrated, ranging from semantic scene understanding to the incorporation of artificial landmarks such as fiducial markers [150].

Beyond enriching maps with visual and depth information, various methodologies aim to organize this data into interpretable and structured representations. Among them, *scene graphs* represent environments as hierarchical structures that capture the presence of objects, their attributes, and inter-relationships. These solutions provide a higher-level abstraction for scene understanding by outlining spatial associations among observed entities [161]. While scene graph-driven works like [251, 160] focus on tailoring geometric and semantic cues for reliable interpretation, approaches such as Hydra [157] and HOV-SG [158] serve as 3D scene graph builders (rather than SLAM systems), representing spatial hierarchies without estimating camera trajectories. Moreover, HOV-SG requires ground-truth semantics and an entire-environment map to construct the scene graph offline. In contrast, works like *S-Graphs* [156] push the boundaries by directly incorporating scene graphs into the SLAM pipeline, relying on LiDAR odometry with planar surface extraction within a unified optimization system, though without support for visual input.

Inspired by LiDAR-based *S-Graphs* [156], this paper proposes a VSLAM framework,

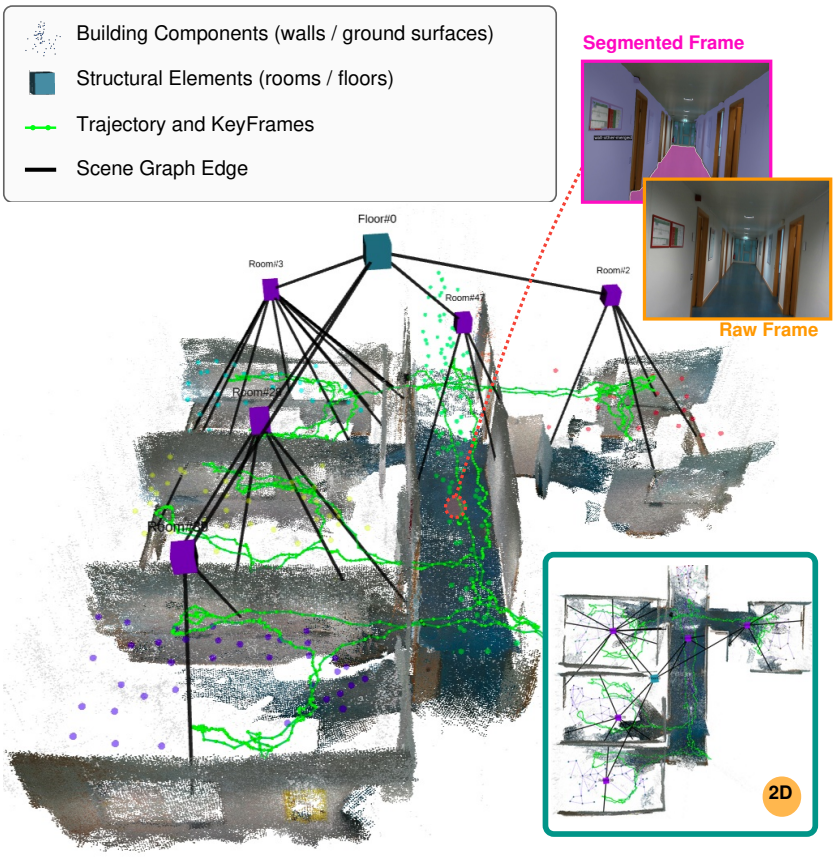


Figure 8.1: A reconstructed map tailored to the optimizable 3D scene graph generated by the proposed vS-Graphs, enriched with environment-driven semantic entities. Distinct color point clouds represent different building components (sequence *MR03* of the *AutoSense* dataset).

visual S-Graphs (vS-Graphs), that seamlessly integrates scene graph generation into the SLAM process. vS-Graphs operates in real time and employs both visual and depth data to enhance map reconstruction and camera pose estimation. It reliably incorporates "building components" (*i.e.,* wall and ground surfaces), "structural elements" (*i.e.,* rooms and floors), and their spatial associations to produce a more structured and semantically coherent environmental representation. Unlike LiDAR-based S-Graphs, which derive semantic–relational structures from geometric data, vS-Graphs exploits the richer semantic and appearance information available in visual data to both validate and contextualize structural elements. This not only strengthens category verification but also establishes a scalable foundation for incorporating additional components in future extensions. The framework also generates interpretable 3D scene graphs with hierarchical optimization capabilities that pair robot poses from the underlying SLAM with detected entities, as depicted in Fig. 8.1. With this, the contributions of the paper are summarized below:

- A real-time multi-threaded VSLAM framework that constructs optimizable 3D scene graphs during map reconstruction,
- A vision-based method for recognizing and mapping building components (*i.e.,* wall and ground surfaces), enhancing map richness and trajectory estimation,
- A mechanism for extracting high-level structural elements from the localized build-

ing components for advancing scene understanding; and

- Publicly available source code to facilitate reproducibility and further research in the field.

8.3 Related Works

Recent advances in computer vision and the development of reliable VSLAM frameworks, such as ORB-SLAM 3.0 [108], have enabled more robust localization and mapping systems. As noted by [166], depth information is essential for enhancing scene understanding and supporting downstream tasks such as environment modeling. In this regard, ElasticFusion [138] constructs globally consistent surfel maps for precise photometric tracking, but struggles with scalability and limited integration of environment-level cues. BAD SLAM [252] improves map and trajectory optimization via direct RGB-D Bundle Adjustment (BA), yet remains sensitive to initialization errors. More recent systems, like GS-SLAM [139], RTG-SLAM [140], and SplaTAM [141], incorporate 3D Gaussian scene representations to enhance tracking and map reconstruction. Nonetheless, they overlook the scene's semantic context.

Building on these advances, RGB-D VSLAM frameworks have begun integrating semantic awareness into their mapping pipelines. Tools like Voxblox++ [253] augment online scanning with volumetric, object-centric mapping for better scene element recognition. NICE-SLAM [142] combines neural implicit and hierarchical representations with pre-trained geometric priors to improve dense scene reconstruction at the cost of heavy computation and increased localization errors. To address semantic awareness, works like SaD-SLAM [133], OVD-SLAM [254], DynaSLAM [129], Dynamic SLAM [132], and YDD-SLAM [255] use Convolutional Neural Networks (CNNs) to filter feature points associated with specific semantic objects, refining pose estimation and trajectory accuracy. Similarly, [256] introduces a binary CNN-based descriptor for robust feature matching while improving the initial pose measurements. Despite these efforts, most systems remain limited to object-level reasoning rather than constructing comprehensive scene representations enriched with environment-driven entities.

Another research direction explores fiducial markers as reliable landmarks, offering an alternative to purely vision-based scene interpretation. In contrast to approaches like UcoSLAM [150], TagSLAM [152], and sSLAM [151], which utilize markers for assembling geometric maps, Approaches like [51, 52] detect and map environment-driven entities labeled with markers, thereby contributing to a more comprehensive understanding of the environment layout. Nevertheless, their reliance on pre-placed markers restricts their applicability to controlled environments, limiting flexibility in unprepared or dynamic settings.

In contrast, our approach introduces a real-time VSLAM framework that unifies scene graph construction with the VSLAM process. Relying on RGB-D data, it more effectively exploits visual and depth cues to enhance mapping accuracy and trajectory estimation, while generating semantically enriched environment representations without external landmarks.

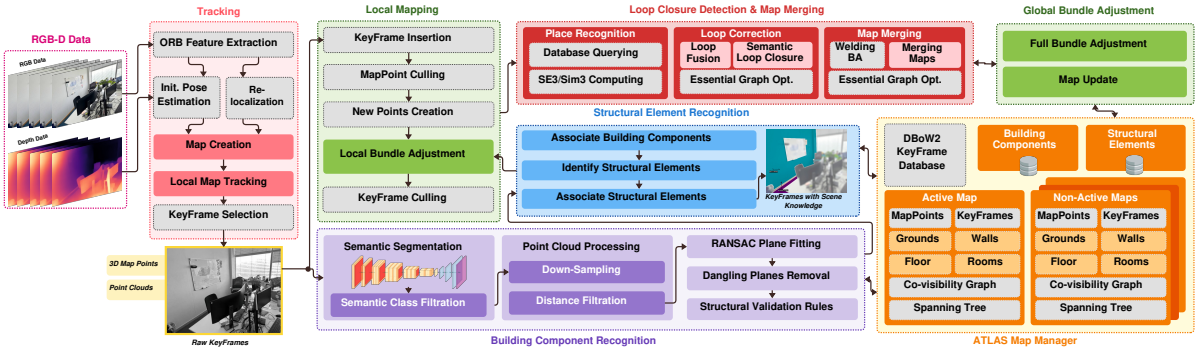


Figure 8.2: The multi-thread architecture of vS-Graphs. Modules with dashed borders and a light gray background are inherited directly from the baseline (*i.e.,* ORB-SLAM 3.0), while the remaining components are newly added or modified modules.

8.4 Proposed Method

8.4.1 System Overview

Building upon ORB-SLAM 3.0, vS-Graphs introduces substantial modifications to its baseline's core modules and adds new threads for robust scene analysis and reconstruction. The system architecture, shown in Fig. 8.2, details the individual threads, components, and their interconnections. The current version supports RGB-D input, primarily using depth data to generate surface point clouds, which are then validated for robust scene understanding. The core contribution lies in seamlessly integrating two novel threads: *Building Component Recognition* (§8.4.2) and *Structural Element Recognition* (§8.4.3). These threads are deeply embedded within vS-Graphs, working tightly with other modules to enrich the reconstructed maps and ensure optimal performance.

In brief, RGB-D data is processed in real time, providing integrated visual and depth information to subsequent modules. Visual features are extracted and tracked across frames in the *Tracking* thread, where pose information is either initialized or refined, depending on the map reconstruction stage, thereby creating a 3D map of tracked features. KeyFrame selection, a critical step following feature extraction, is performed within the *Tracking* thread by analyzing the visual data. These KeyFrames contain 3D map points and point clouds, forming the foundation for subsequent processes.

KeyFrames are then sent to the *Local Mapping* thread for map integration and optimization, with inaccurately posed KeyFrames culled for enhanced accuracy. Simultaneously, the *Building Component Recognition* thread identifies and localizes walls and ground surfaces by processing the KeyFrame-level visual-spatial data. These components are precisely detected by filtering the data through a CNN, followed by Random Sample Consensus (RANSAC) [79] plane fitting for global pose estimation. Concurrently, the *Structural Element Recognition* thread runs at fixed intervals, extracting higher-level entities, including rooms and floors, from the building components of the current (active) map. Eventually, the *local bundle adjustment* in the *Local Mapping* thread optimizes KeyFrame poses and map points for local consistency, as well as refining building components and structural elements, jointly optimizing geometric and semantic entities for accurate map reconstruction.

In the baseline, the *Atlas* map manager serves as the core of the multi-map management procedure, maintaining mapped instances and their associations in an active

map, and supporting operations such as map merging. In vS-Graphs, we extend this core to store higher-level semantic entities, including building components and structural elements, enriching the map representation and extending storage and retrieval beyond purely geometric information. Furthermore, the *Loop Closure Detection* checks for revisited locations and triggers *Global Bundle Adjustment* for optimization and map merging when a loop is detected. While the loop closure mechanism remains fundamentally geometric, the inclusion of environment-driven entities enables the system to verify and refine loop candidates. During loop closure and global optimization, redundant building components and structural elements detected across overlapping KeyFrames are automatically merged and their associations updated, ensuring a consistent map representation. Although the *semantic loop closure* module provides an additional consistency check that complements geometric matching, a fully semantic-driven loop closure triggered solely by the association of the detected building components and structural elements is an area for future investigation.

8.4.2 Building Component Recognition

This thread processes the point cloud and visual data from KeyFrames to extract fundamental environment-driven elements beneficial for scene understanding. The current version of vS-Graphs defines building components $\mathbf{\Psi}$ as walls and ground surfaces. Accordingly, each KeyFrame $K = \{\mathbf{P}, L, \varepsilon\}$ passes through the *building component recognition* module B, generating identified structural element set $\mathbf{\Pi}^K_{\mathbf{\Psi}} = B(\mathbf{K})$ for K. In the KeyFrame level, $\mathbf{P} = \{p_i \mid p \in \mathbb{R}^3, i \in \mathbb{N}\}$ is the point cloud, L is the matrix of RGB data, and ε represents additional metadata for mapping (such as camera pose).

Building component extraction is achieved by processing L through a semantic segmentation function S, which labels each pixel in the RGB data, retaining only the relevant classes and discarding irrelevant entities. In contrast to conventional object detection methods, panoptic scene segmentation provides *pixel-level class labels* and *instance-level differentiation*, enabling sharper boundary delineation and more precise object recognition. In principle, any reliable real-time panoptic segmentation framework can be integrated into vS-Graphs. However, the current version has integrated Panoptic-FCN (pFCN) [257] and YOSO [258] by default, due to their efficiency and strong balance between accuracy and real-time performance. In this regard, $\widehat{\mathbf{P}}^K_{\mathbf{\Psi}} = S(\mathbf{P}, L)$ refers to the segmented visual-spatial data of K, where $L_{(u,v)}$ refers to the semantic class label of a pixel at $(u, v) \in \mathbb{N}^2$. Applying semantic class filtration to filter building components' semantic classes $\mathbf{\Psi}$ (*i.e.,* wall and ground surfaces) takes place as follows:

$$\widehat{\mathbf{P}}^K_{\psi} = \{p_j \mid p_j \in \mathbb{R}^3, L_{p_j} \in \mathbf{\Psi}\} \tag{8.1}$$

$$\widehat{\mathbf{P}}^K_{\mathbf{\Psi}} = \{\widehat{\mathbf{P}}^K_{\psi}\} = \{\widehat{\mathbf{P}}^K_{wall}, \widehat{\mathbf{P}}^K_{ground}\} \tag{8.2}$$

where $\widehat{\mathbf{P}}^K_{\mathbf{\Psi}} \subset \mathbf{P}$ represents the set of semantically segmented point clouds, including the point clouds of $\widehat{\mathbf{P}}^K_{wall}$ and $\widehat{\mathbf{P}}^K_{ground}$. It should be noted that the classification confidence $\lambda_{(u,v)} \in \mathbb{R}$ for each pixel $L_{\Psi(u,v)}$ is set for potential classification errors.

The next stage is to optimize $\widehat{\mathbf{P}}^K_{\mathbf{\Psi}}$, as it may contain noisy or low-resolution points that negatively impact subsequent steps. In this regard, each segmented visual-spatial point cloud $\widehat{\mathbf{P}}^K_{\psi}$ undergoes a two-stage preprocessing procedure. The first step is applying downsampling to fetch $\widehat{\mathbf{P}}^K_{\psi_d}$, a refined point cloud with reduced redundancy and noise, where $\widehat{\mathbf{P}}^K_{\psi_d} \subset \widehat{\mathbf{P}}^K_{\psi}$. As depth sensors exhibit increased noise at extreme ranges, the semantic

classification of structural elements is most reliable within an optimal depth range. Thus, applying distance filtration based on the recommended sensor range results in retaining a subset of the point cloud as $\widehat{\mathbf{P}}^K_{\psi_\zeta} \subset \widehat{\mathbf{P}}^K_{\psi_d}$. The final processed point cloud, referred to as $\widehat{\mathbf{P}}^K_{\mathbf{\Psi}_\zeta} = \{\widehat{\mathbf{P}}^K_{\psi_\zeta}\}$, is then forwarded to successive stages for further analysis.

Processing each $\widehat{\mathbf{P}}^K_{\psi_\zeta}$ through the RANSAC plane fitting algorithm results in detecting semantically-validated building components with their geometric equations. Thus, sets of random points $\mathbf{p_r} \in \widehat{\mathbf{P}}^K_{\psi_\zeta}$ are iteratively selected to calculate the normal vectors $\mathbf{n} \in \mathbb{R}^3$ representing validated planar components π with the pre-defined distance $d(\mathbf{p_r}, \pi) \leq \epsilon$, where ϵ is the inlier threshold. Accordingly, the final output representing all defined building components is as follows:

$$\mathbf{\Pi}^K_\psi = \{{\pi_\psi}_i^K \mid {\pi_\psi}_i^K \in \mathbb{R}^3, i \in \mathbb{N}\} \tag{8.3}$$

$$\mathbf{\Pi}^K_{\mathbf{\Psi}} = \{\mathbf{\Pi}^K_\psi\} = \{\mathbf{\Pi}^K_{wall}, \mathbf{\Pi}^K_{ground}\} \tag{8.4}$$

The remaining reliable elements are checked against structural validation rules $\mathbf{\Pi}^K_{\mathbf{\Psi}} = V(\mathbf{\Pi}^K_{\mathbf{\Psi}})$, ensuring that they satisfy reasonable geometric constraints for the environment. For instance, walls should be represented as vertical planes, while ground surfaces should be defined as horizontal planes. The thread concludes the processing of KeyFrame K by storing its detected building components $\mathbf{\Pi}^K_{\mathbf{\Psi}}$ within the current map in the *Atlas*, ensuring they contribute to the ongoing map reconstruction.

8.4.3 Structural Element Recognition

This thread repeatedly runs at constant time intervals (*i.e.,* every two seconds) to detect potential higher-level semantic entities that characterize the environment's layout, including rooms and floors. Structural elements comprise multiple building components, with their topological associations accounted for. The thread actively searches for layouts that form rooms, where a room is a spatially enclosed area bounded by at least two walls that surround a confined free-space cluster on the ground plane. Additionally, a floor represents a higher-order structural element that encompasses a collection of rooms within a single building level.

In the first step, the map's existing building components $\mathbf{\Pi}_{\mathbf{\Psi}} = \{\mathbf{\Pi}^{K_i}_{\mathbf{\Psi}} \mid i \in \mathbb{N}\}$ are fetched from *Atlas*. Associating and merging redundant or conflicting building components is crucial to ensure consistency in the map reconstruction. In this context, the primary factors to assess are the spatial proximity and alignment of the mapped building components, requiring evaluating both the Euclidean distance and the angular difference between their normal vectors. Accordingly, the building component association condition is presented below:

$$\|\mathbf{\Pi}^{K_i}_{\mathbf{\Psi}} - \mathbf{\Pi}^{K_j}_{\mathbf{\Psi}}\| \leq \rho \quad \wedge \quad \cos^{-1}\left(\frac{\mathbf{n}_i \cdot \mathbf{n}_j}{\|\mathbf{n}_i\|\|\mathbf{n}_j\|}\right) \leq \eta \tag{8.5}$$

where $\mathbf{n}$ is the normal vector of a particular surface and ρ and η refer to spatial proximity and angular alignment thresholds, respectively. The associated building components $\mathbf{\Pi}_{\mathbf{\Psi}}$ are subsequently utilized to detect structural elements $\mathbf{\Delta}_{\mathbf{\Phi}}$, defined as below:

$$\mathbf{\Delta}^K_\phi = \{{\delta_\phi}_m^K \mid {\delta_\phi}_m^K \in \mathbb{R}^3, m \in \mathbb{N}\} \tag{8.6}$$

$$\mathbf{\Delta}^K_{\mathbf{\Phi}} = \{\mathbf{\Delta}^K_\phi\} = \{\mathbf{\Delta}^K_{room}, \mathbf{\Delta}^K_{floor}\} \tag{8.7}$$

where $\mathbf{\Delta}_{\mathbf{\Phi}}^{K} \in \mathbf{\Delta}_{\mathbf{\Phi}}$ is the set of structural elements belong to semantic classes $\mathbf{\Phi}$ found in KeyFrame K. The procedure for structural element detection is outlined below.

Rooms. A room $\delta_r^K = \{\mathbf{\Pi}_{wall}, \pi_{ground}^{K_q}, \nu_r\}$ is a convex, spatially enclosed area bounded by at least two walls, each oriented toward a distinct free-space cluster $\mathbf{\Upsilon} = \{\upsilon_j \mid \upsilon_j \in \mathbb{R}^3, j \in \mathbb{N}\}$ in the global frame. The free-space cluster is calculated from the depth camera points using the Voxblox tool [259]. The associated set of walls is denoted as $\mathbf{\Pi}_{wall} = \{\pi_{wall}^{K_1}, \ldots, \pi_{wall}^{K_n}\}$, where $n \geq 2$. Each wall $\pi_{wall}^{K_i}$ is validated using *proximity* and *directionality* constraints, ensuring geometric consistency with its corresponding free-space. *Proximity* is calculated as:

$$\forall\, \pi_{wall}^{K_i} \in \mathbf{\Pi}_{wall},\ \exists\, \upsilon_j \in \mathbf{\Upsilon} :\ \left| \mathbf{n}_{wall}^{K_i \top} \cdot \upsilon_j + d_{wall}^{K_i} \right| \leq \omega \tag{8.8}$$

where $\mathbf{n}_{wall}^{K_i} \in \mathbb{R}^3$ denotes the wall's normal vector, $d_{wall}^{K_i}$ is the plane offset, and ω is a tunable threshold controlling the allowable distance between the wall and nearby free-space points. *Directionality* is computed as follows:

$$\forall \pi_{wall}^{K_i} \in \mathbf{\Pi}_{wall} :\ \mathbf{n}_{wall}^{K_i \top} \left(\nu_{\Upsilon} - \nu_{wall}^{K_i} \right) < 0 \tag{8.9}$$

where $\nu_{wall}^{K_i} \in \mathbb{R}^3$ is the wall centroid and $\nu_{\Upsilon} \in \mathbb{R}^3$ is the centroid of the associated free-space cluster Υ. These formulations enable the detection of convex n-wall rooms with arbitrary wall orientations, beyond Manhattan layouts, by enforcing that adjacent walls face the enclosed cluster. The ground plane $\pi_{ground}^{K_q}$ is associated with a room if it is approximately orthogonal to all wall normals and spatially enclosed by them. Accordingly:

$$\left| \mathbf{n}_{ground}^{\top} \cdot \mathbf{n}_{wall}^{K_i} \right| \leq \sin(\vartheta) \quad \wedge \quad d_{\pi_{wall}^{K_i}} \leq \nu_g \leq d_{\pi_{wall}^{K_j}} \tag{8.10}$$

where ϑ defines the angular tolerance and ν_g refers to the ground centroid lying between the nearest and farthest walls.

The optimization incorporates a geometry-aware constraint mechanism that updates each room's centroid (νr_i) based on the centroids of its associated walls, while simultaneously identifying *parallel* and *perpendicular* wall pairs within the same room. Such patterns are naturally common in man-made environments, where walls are commonly constructed to align along dominant structural axes. These relationships introduce geometric constraints that minimize angular deviations, encouraging walls to align closer to 0° or 90°, respectively. In the absence of such relations, only the centroid consistency term remains active, maintaining spatial coherence without enforcing unnecessary structure. This flexible design enables the system to accommodate rooms of arbitrary or asymmetric geometry without relying on prior layout assumptions, unlike rectangular-shaped formulations commonly used in previous works [155, 157, 158]. In this context, if a room contains parallel walls facing each other toward ν_{Υ}, a parallelism constraint penalizes deviation through a cost given by the angle between their normals:

$$c^{\parallel}_{\delta_r^K (i,j)} = 1 - \left| \mathbf{n}_i^{\top} \cdot \mathbf{n}_j \right|, \quad \text{where } |\mathbf{n}_i| = |\mathbf{n}_j| = 1. \tag{8.11}$$

where $\mathbf{n}_i$ and $\mathbf{n}_j$ are the normalized values of normal vectors of the two wall planes. The cost reaches zero for perfectly parallel walls ($\mathbf{n}_i \parallel \mathbf{n}_j$) and increases proportionally with angular deviation, providing a numerically stable measure of parallelism. Similarly, perpendicularity is calculated as:

$$c^{\perp}_{\delta_r^K (i,j)} = \left| \mathbf{n}_i^{\top} \cdot \mathbf{n}_j \right|, \quad \text{where } |\mathbf{n}_i| = |\mathbf{n}_j| = 1. \tag{8.12}$$

Finally, a centroid consistency term c_{ν_r} keeps the room centroid ν_r aligned with the mean of its wall centroids. The overall cost function for room-level constraints is:

$$c_{\nu_r} = \|\hat{\nu_r} - \frac{1}{n}\sum_{j=1}^{n} \nu_{wall}^{K_j}\|^2 \tag{8.13}$$

$$c_{\delta_r^K} = \frac{1}{N_\parallel}\sum_{i,j=0}^{N_\parallel} c^{\parallel}_{\delta_r^K(i,j)} + \frac{1}{N_\perp}\sum_{i,j=0}^{N_\perp} c^{\perp}_{\delta_r^K(i,j)} + c_{\nu_{r_i}} \tag{8.14}$$

were $N_\parallel$ and $N_\perp$ denote the number of parallel and perpendicular walls. Hence, when no parallel or perpendicular relations are detected, only the Euclidean-norm–based centroid update is activated, preserving localization accuracy while enhancing the structural coherence of the 3D scene graph.

Floors. A floor $\delta_f^K = \{\boldsymbol{\Delta}_r, \nu_f, \pi_{floor}^{K_i}\}$ is defined by a set of rooms $\boldsymbol{\Delta}_r$, all sharing a common horizontal reference plane $\pi_{floor}^{K_i}$ and the floor centroid ν_f. The floor plane is associated with the constituent rooms if it satisfies the co-planarity condition within a vertical tolerance. In practice, this validation step is ignored in the current vS-Graphs implementation due to its single-floor design constraint. Additionally, its spatial extent is bounded by the union of all constituent room boundaries. The floor centroid ν_f is computed as the weighted arithmetic mean of all constituent room centroids:

$$\nu_f = \frac{\sum_{j=1}^{N_r} \nu_{r_j}}{N_r} \tag{8.15}$$

where $N_r = |\boldsymbol{\Delta}_{room}|$. The cost function to optimize the floor's vertex node is calculated as follows:

$$c_{\delta_f^K} = \sum_{t=1}^{T} \left\|\hat{\nu}_f - h(\boldsymbol{\Delta}_r^K)\right\|^2_{\boldsymbol{\Lambda}_{\tilde{\delta}_{f,t}}} \tag{8.16}$$

where $h(\dots)$ is a hierarchical mapping function that associates the floor's structural components with its centroid.

8.4.4 Scene Graph Structure

Fig. 8.3 depicts the 3D scene graph structure generated using vS-Graphs, along with its corresponding modules introduced in Fig. 8.2. In contrast to the traditional SLAM reconstructed maps, the geometric replicas of vS-Graphs are augmented with hierarchical, rich semantic data, enabling meaningful interaction with the environment. The generated scene graph fills the gap in contextual understanding and enables better scene interpretation, enabling complementary missions such as scalable map-building and improved decision-making.

8.5 Experimental Results

8.5.1 Methodology

Setup. Evaluations were conducted on a system with an Intel® Core™ i9-11950H processor (2.60 GHz), a 4GB NVIDIA T600 Mobile GPU, and 32GB of RAM.

Datasets. We evaluated vS-Graphs on standard benchmarks and in-house datasets to assess its performance across diverse environments and scene complexities. The standard datasets include ICL [207] photorealistic synthetic scenes and real-world data from OpenLORIS [205], ScanNet [260], and TUM-RGBD [167]. The in-house dataset, *AutoSense*,

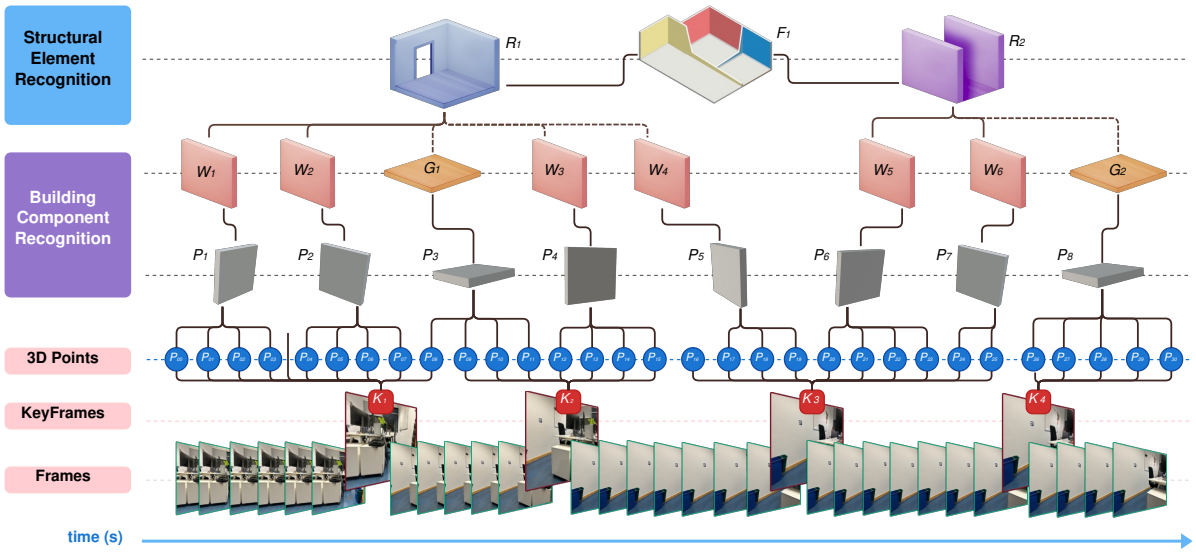


Figure 8.3: Scene graph structure generated using vS-Graphs, creating a hierarchical representation of the environment.

was collected using a custom-built device that records RGB-D video and LiDAR point clouds. It features diverse real-world indoor environments with varied architectural layouts and multi-room scenarios. The ground truth data in the dataset was derived from reliable LiDAR poses and point clouds generated by *S-Graphs*. Table 8.1 outlines the dataset characteristics, and Fig. 8.4 showcases the device and sample sequences. Due to space constraints, the complete evaluation results and figures are available at https://snt-arg.github.io/vsgraphs-results/.

8.5.2 Evaluations and Discussions

Trajectory Estimation Performance

To assess trajectory estimation accuracy, vS-Graphs was benchmarked against established and robust VSLAM systems, including ORB-SLAM 3.0 (baseline) [108], ElasticFusion [138], and BAD SLAM [252]. Marker-based methods (*e.g.,* [51]) were excluded due to their reliance on artificial landmarks and external pose constraints, limiting applicability

Table 8.1: Characteristics of the *AutoSense* in-house dataset. Each sequence includes one ground and one floor surface by design. *BC* denotes building components (*ground/walls*), and *SE* denotes structural elements (*rooms/floor*).

Seq.	#BC *(Ground+Walls)*	#SE *(Room+Floor)*	Description
SR01	4 (1+3)	2 (1+1)	partial view of a room
SR02	4 (1+3)	2 (1+1)	room corner with intersecting walls
SR03	6 (1+5)	2 (1+1)	a single rectangular room
MR01	14 (1+13)	4 (3+1)	across rooms connected to a corridor
MR02	13 (1+12)	4 (3+1)	adjoining rooms linked by a corridor
MR03	22 (1+21)	6 (5+1)	a suite of interconnected rooms

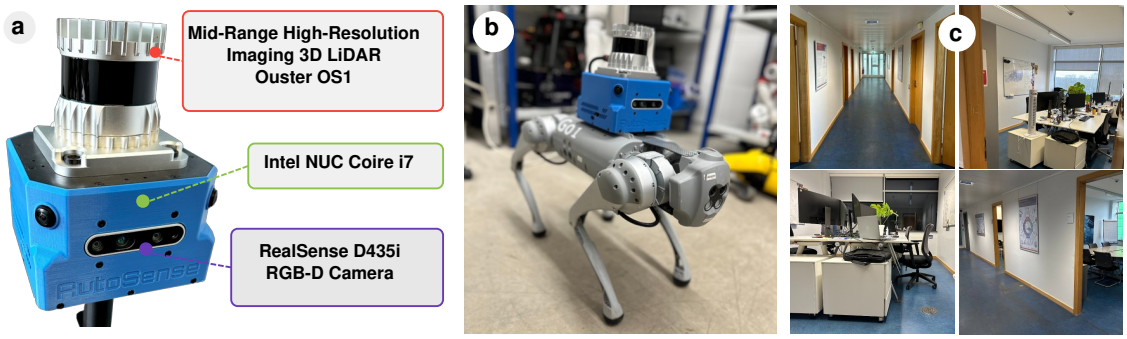


Figure 8.4: In-house dataset collection using the *AutoSense* device: **a)** the setup overview, **b)** the device mounted on a legged robot, and **c)** some instances of the collected data.

in marker-free environments. Similarly, neural field-based approaches (such as [142]) were omitted because they rely on learned scene priors rather than explicit geometric–semantic representations. Additionally, vS-Graphs was evaluated with different segmentation backbones (pFCN and YOSO) to study the effect of segmentation quality and efficiency on trajectory robustness.

Table 8.2 presents the evaluation results, where each system is evaluated over eight runs per sequence, and performance is measured using Absolute Trajectory Error (ATE) reported in centimeters. Dashes in Table 8.2 indicate unavailable data due to failure in tracking. Accordingly, vS-Graphs variants consistently achieve state-of-the-art performance, ranking first or second in nearly all evaluations across segmentation backbones and module configurations. This improvement stems from the additional geometric and semantic constraints imposed by accurately localized building components and structural elements. When combined with loop closures, these constraints yield even larger gains, *e.g.,* 75.38% for "*deer-w*" with the YOSO-based configuration. However, rapid motion and noisy depth data can negatively affect performance (*e.g.,* "*SR01*", BC-only). Another notable observation is that incorporating room entities further improves trajectory accuracy, particularly when room–wall constraints are enforced. This effect is most evident in looped ("*deer-w*") and multi-room environments ("*MR01*"), where maintaining geometric consistency is crucial, reducing ATE by 13.82% and 15.22% w.r.t. the baseline. It can be seen that the choice of panoptic backbone (YOSO or pFCN) has only a marginal effect on the trajectory accuracy of vS-Graphs, with higher segmentation quality naturally yielding more precise structural mapping.

It should be noted that a primary challenge shared by all tested VSLAMs arises in low-texture scenes (*e.g.,* corridors with uniformly painted walls), negatively impacting feature matching and tracking. In vS-Graphs, this can affect pose estimation and localization of building components and propagate the error through the structural element recognition stage. Additionally, treating building components as planar surfaces might limit the framework's applicability in environments with irregular geometries (*e.g.,* curved walls). Furthermore, overly or loosely permissive association thresholds can degrade building component identification performance, making careful threshold selection crucial.

Mapping Performance

Additionally, analyzing the accuracy of the reconstructed maps against *AutoSense*'s ground truth data revealed that vS-Graphs performs more robustly compared to ORB-SLAM 3.0 in terms of Root Mean Square Deviation (RMSE). As shown in Fig. 8.5, the median

Table 8.2: Absolute Trajectory Error (ATE) in centimeters (cm) for various VSLAM algorithms across eight iterations. The best and second-best results are **boldfaced** and underlined, respectively. vS-Graphs is evaluated with multiple semantic segmentation backbones (YOSO and pFCN) and with different levels of entity integration, including only building component cues (*vS-Graphs BC*) and the full pipeline with structural element recognition (*vS-Graphs*). The minimum completion rate of each sequence is set to 50%, and dashes indicate unavailable data due to tracking failures. Additionally, the "*%Gain*" rows quantify the percentage improvement achieved by vS-Graphs compared to its baseline (ORB-SLAM 3.0).

Dataset	Sequence	Time	Methodology										
						vS-Graphs				vS-Graphs vs. baseline			
						BC Only		*BC+SE*		*BC Only*		*BC+SE*	
			ORB-SLAM3	BAD SLAM	ElasticFusion	YOSO	pFCN	YOSO	pFCN	YOSO	pFCN	YOSO	pFCN
ICL	deer-ground	79.9	**0.74**	147.60	14.50	0.81	0.91	0.78	0.89	-8.89	-23.40	-5.25	-20.09
	deer-walkh	65.3	6.96	–	82.50	6.08	6.23	**5.96**	6.14	12.64	10.54	14.38	11.80
	deer-walk	64.0	9.91	147.40	62.00	8.82	12.19	**2.44**	4.02	10.97	-22.98	75.38	59.43
	deer-running	28.4	6.92	–	78.70	5.34	5.47	**2.83**	3.34	22.82	20.98	59.12	51.75
	deer-mav-fast	102.5	2.67	4.60	–	2.65	2.07	2.23	**1.26**	0.96	22.57	16.61	52.88
	Total	340	5.44	99.87	59.43	4.74	5.37	**2.85**	3.13	7.70	1.54	32.05	31.15
OpenLORIS	office1-1	27.0	6.37	12.30	6.88	6.41	**6.33**	6.42	6.37	-0.66	0.50	-0.85	-0.06
	office1-2	30.0	10.05	14.00	11.26	9.91	9.93	**9.28**	9.68	1.39	1.24	7.67	3.69
	office1-3	12.0	14.73	19.50	–	14.57	16.05	**14.44**	15.01	1.08	-9.00	1.94	-1.93
	office1-4	29.0	12.51	34.30	18.03	12.68	12.55	11.54	**11.36**	-1.37	-0.33	7.77	9.21
	office1-5	53.0	11.57	30.20	22.15	11.11	11.29	10.71	**10.59**	4.03	2.48	7.45	8.49
	office1-6	36.5	5.86	10.70	–	5.90	5.89	5.70	**5.67**	-0.68	-0.45	2.76	3.27
	office1-7	38.6	17.17	20.30	–	17.55	17.50	**13.00**	13.25	-2.17	-1.92	24.30	22.84
	Total	226.1	11.18	20.19	14.58	11.16	11.36	**10.16**	10.28	0.23	-1.07	7.29	6.50
ScanNet	scn0041_01	75.0	14.26	22.20	21.96	14.29	14.06	**13.56**	13.78	-0.27	1.37	4.88	3.34
	scn0200_00	37.0	4.88	–	–	4.67	4.65	4.55	**4.42**	4.36	4.79	6.85	9.52
	scn0614_01	36.2	13.55	13.90	20.23	12.50	13.38	**12.36**	12.45	7.75	1.26	8.76	8.10
	scn0626_00	21.1	9.60	27.30	18.91	9.50	9.22	8.73	**8.71**	1.05	4.02	9.07	9.28
	Total	169.3	10.57	21.13	20.36	10.24	10.33	**9.80**	9.84	3.22	2.86	7.39	7.56
TUM-RGBD	frb1-desk	23.8	2.07	2.20	2.55	1.98	**1.96**	2.04	2.03	4.25	5.18	1.44	1.93
	frb1-desk2	25.1	3.23	2.90	7.87	3.19	3.19	2.89	**2.85**	1.17	1.21	10.50	11.74
	frb1-room	49.1	13.25	20.90	16.75	13.02	13.29	**5.13**	6.77	1.75	-0.26	61.29	48.91
	frb2-desk-prs	142.1	1.90	9.60	4.51	1.79	**1.78**	1.79	1.81	5.69	6.07	5.80	4.75
	frb3-strct	31.9	1.62	2.10	1.75	1.54	1.53	**1.52**	1.54	5.04	5.64	6.32	5.08
	Total	272	4.41	7.54	6.68	4.31	4.35	**2.67**	3.00	3.58	3.57	17.07	14.48
AutoSense	SR01	61	7.98	13.00	**6.62**	8.10	8.16	7.34	7.63	-1.58	-2.28	7.99	4.36
	SR02	80	8.75	13.50	–	**8.21**	8.29	8.23	8.26	6.15	5.32	5.95	5.60
	SR03	170	10.90	55.70	–	10.70	10.71	**10.51**	11.04	1.85	1.71	3.56	-1.30
	MR01	305	15.26	41.30	–	15.44	14.98	**12.23**	12.75	-1.13	1.85	19.87	16.47
	MR02	210	23.97	–	53.48	21.68	23.78	**20.13**	21.19	9.58	0.83	16.03	11.61
	MR03	783	22.19	81.21	–	22.00	19.23	**17.70**	17.84	0.87	13.35	20.24	19.61
	Total	1609	14.84	40.94	30.05	14.35	14.19	**12.69**	13.12	2.63	3.46	12.27	9.39
Overall Total		43m 37s	9.29	37.93	26.22	8.96	9.12	**7.63**	7.87	3.47	2.07	15.22	13.82

RMSE is consistently lower in vS-Graphs, indicating a higher level of overall mapping precision. vS-Graphs achieves superior mapping accuracy despite generating maps with $\sim$ 10.15% fewer points on average than the baseline, owing to its environment-driven constraints that enable a more coherent reconstruction.

Scene Understanding Performance

This experiment evaluates vS-Graphs for semantic scene understanding, with a particular focus on detecting entities essential to interpreting the environment's structural layout.

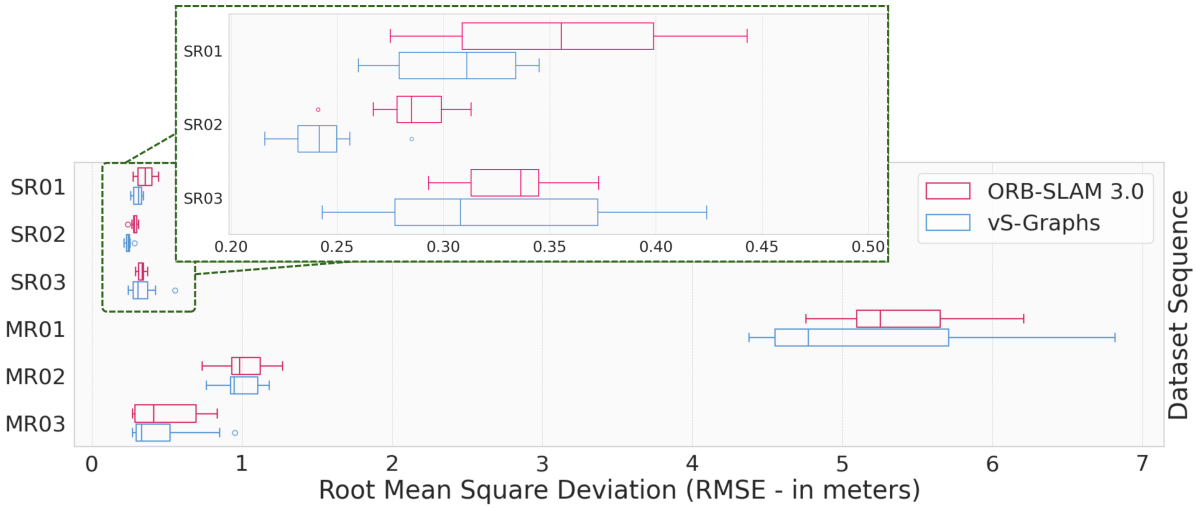


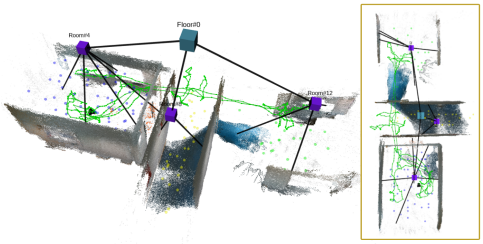
Figure 8.5: Mapping performance across eight iterations, showing *AutoSense* sequences with less than one meter RMSE.

Table 8.3: Scene understanding accuracy of vS-Graphs on multi-room sequences of *AutoSense*. *BC* and *SE* denote "building components" and "structural elements," respectively, while *GT* indicates the ground truth count of manually verified items.

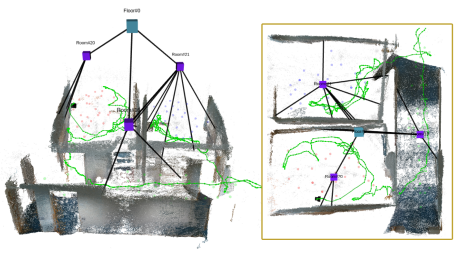
		Detected / GT		**Precision**	**Recall**
Method	*Sequence*	*BC*	*SE*	*BC*	*BC*
S-Graphs [156]	MR01	11 / 14	4 / 4	0.92	0.92
	MR02	12 / 13	4 / 4	1.00	0.92
	MR03	20 / 22	6 / 6	0.90	0.95
Hydra [157]	MR01	N/A	4 / 4	N/A	N/A
	MR02	N/A	5 / 4	N/A	N/A
	MR03	N/A	6 / 6	N/A	N/A
vS-Graphs (ours)	MR01	13 / 14	4 / 4	0.86	1.00
	MR02	13 / 13	4 / 4	0.92	0.92
	MR03	23 / 22	6 / 6	0.96	1.00

For benchmarking, we selected the *multi-room sequences* from the *AutoSense* dataset, which include ground truth annotations derived from LiDAR scans. Table 8.3 reports a quantitative comparison of vS-Graphs against two state-of-the-art approaches: Hydra [157] and *S-Graphs* [156]. For presentation clarity, *precision* and *recall* values for structural elements were omitted from the table to avoid visual saturation. The ground truth counts of rooms and walls were obtained by manual inspection of the dataset, where each entity was visually verified and counted. While *S-Graphs* benefits from the geometric accuracy of LiDAR point clouds, Hydra was configured to use visual point clouds, ensuring a fair comparison against our purely vision-based approach.

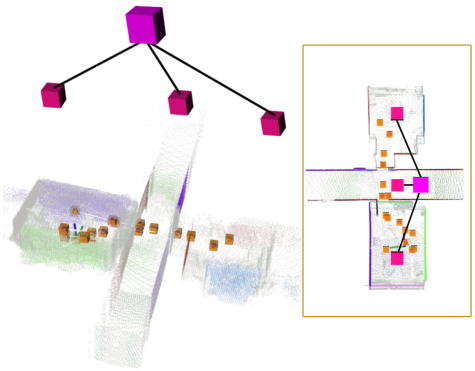
Experimental results show that vS-Graphs, despite operating solely on RGB-D input, achieves an accuracy comparable to the LiDAR-based method (*S-Graphs*) in detecting building components and structural elements. This highlights the effectiveness of our visual feature processing and scene graph generation in capturing structural layouts with



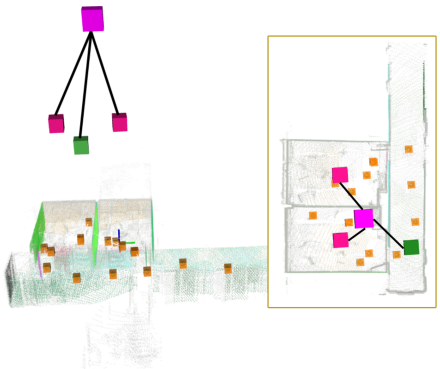
(a) vS-Graphs on *MR01*



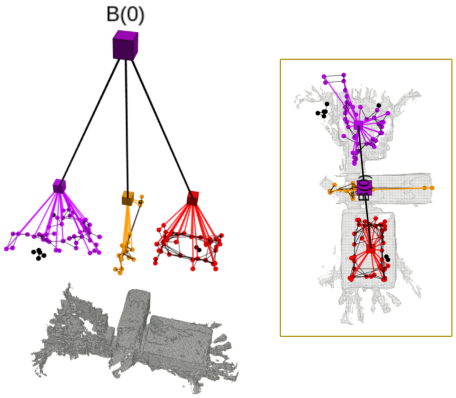
(b) vS-Graphs on *MR02*



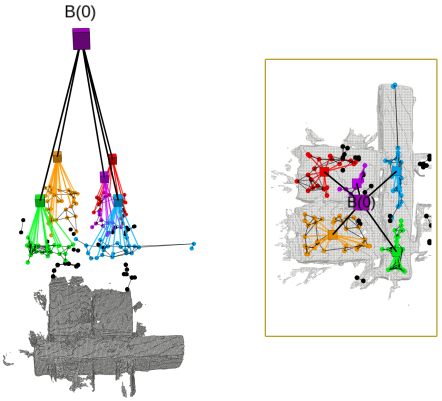
(c) *S-Graphs* on *MR01*



(d) *S-Graphs* on *MR02*



(e) Hydra on *MR01*



(f) Hydra on *MR02*

Figure 8.6: Qualitative comparison of reconstructed scene graphs on multi-room sequences of the *AutoSense* dataset.

high precision. Notably, vS-Graphs demonstrates higher *recall*, even in larger, more complex environments such as MR03, owing to its visual verification of building components, which ensures consistent entity detection across extended scenes. It should be noted that "wall" entities are not directly provided in Hydra; therefore, Hydra's performance is assessed based on correct "room" element counting and recognition. Additionally, Fig. 8.6 provides a qualitative comparison of the reconstructed scene graphs generated by vS-Graphs, *S-Graphs*, and Hydra across two representative dataset sequences.

Runtime Analysis

vS-Graphs achieves real-time performance with an average processing rate of 22 ± 3 FPS, exceeding the 20 FPS threshold considered for real-time operation. This is accomplished through a multi-threaded architecture, as shown in Fig. 8.7. The "Tracking" thread processes visual features at the frame level, while "Local Mapping" concurrently maps objects

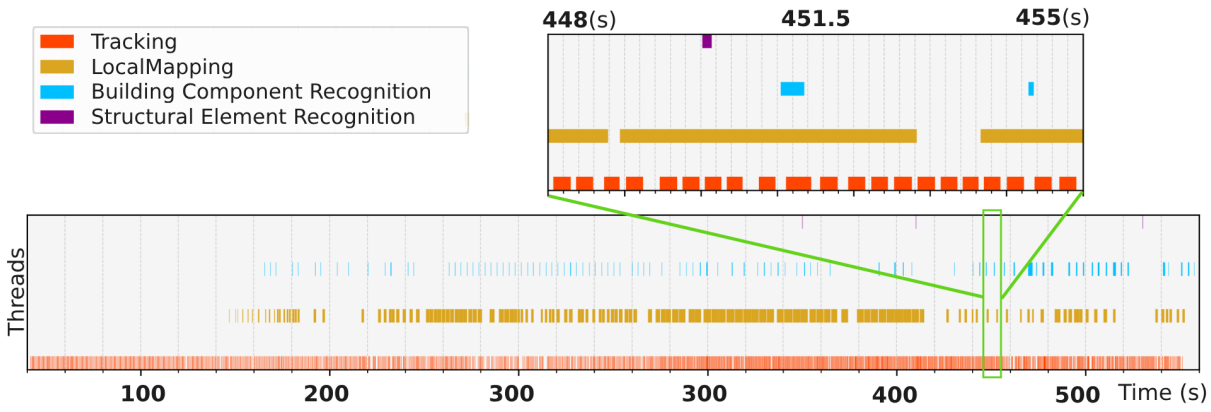


Figure 8.7: Timeline of thread execution within vS-Graphs, while processing a sample dataset instance (sequence *room-walls*).

and optimizes their positions. "Building Component Recognition," running in parallel at the KeyFrame level, identifies potential wall and ground surfaces from the online panoptic segmentation. The "Structural Element Recognition" runs less frequently and in constant periods (every two seconds) to infer rooms and floors in the map. Compared to ORB-SLAM 3.0's 29 ± 3 FPS on the same hardware and dataset, the slightly reduced frame rate is a reasonable trade-off for vS-Graphs's rich semantic scene understanding.

8.6 Conclusions

This paper introduced vS-Graphs, a real-time VSLAM framework that reconstructs the robot's operating environment using optimizable hierarchical 3D scene graphs. It detects building components (wall and ground surfaces), infers structural elements (rooms and floors) from them, and incorporates them into hierarchical representations. Consequently, beyond enhancing map reconstruction by integrating these entities, vS-Graphs offers structured representations of spatial relationships among high-level environment-driven semantic entities. Experimental results on standard and in-house indoor datasets showed that the framework achieves superior trajectory estimation and mapping performance compared to state-of-the-art VSLAMs, reducing trajectory error by 15.22% across a wide range of datasets. Other evaluations showed that the visual features processed by vS-Graphs can identify semantic entities describing the environment's layout with accuracy comparable to precise LiDAR-based methods.

Future work includes integrating additional building components (*e.g.,* ceilings, doorways), extending support for complex and concave room layouts via Graph Neural Network (GNN)-based approaches, and developing fully semantic-driven loop closure detection, where revisited areas can be identified primarily based on previously detected structural elements and building components. Another direction is extending the framework for *multi-floor* setups.

8.7 Appendices

Appendix I. Semantic Augmentation using Fiducial Markers

In vS-Graphs, fiducial markers serve as an **optional semantic augmentation mechanism**, providing high-level contextual information to the reconstructed map. Unlike marker-based Visual SLAM methods [51, 52, 151, 150], where markers play a central role

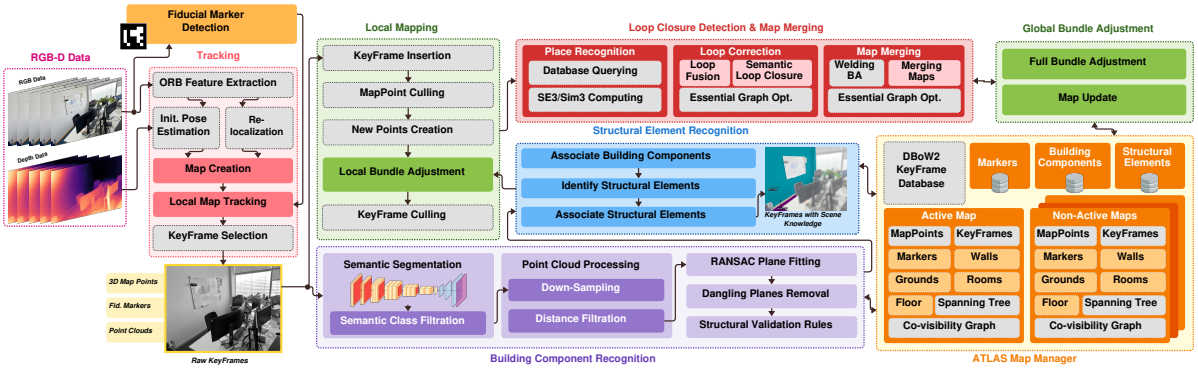


Figure 8.8: The multi-threaded architecture of vS-Graphs with the optional fiducial marker detector module. Modules with dashed borders and a light gray background are inherited directly from the baseline (*i.e.,* ORB-SLAM 3.0), while the remaining components are the contributions of vS-Graphs.

in localization and map construction, markers in our vS-Graphs system serve a supportive rather than structural role. The pipeline does not depend on the presence of markers for map accuracy or robustness; instead, they enrich the representation by linking structural elements to semantic labels. To enable this, we assume that each structural element (*e.g.,* a room) may contain one unique **ArUco marker**. This criterion is performed during data collection, where unique markers are placed in each free-space zone (e.g., rooms or corridors), and their associations are stored in a dedicated environment-driven database (`JSON` file). For instance, a marker with marker_id = 5 is assigned to *corridor-A*, while marker marker_id = 8 was assigned to *office-B*. At runtime, if (i) **the ArUco detector is active** and (ii) **the corresponding association file (database of marker information) is provided**, the system uses this information to enrich detected structural elements semantically. Thus, if the pose of a detected marker falls within the area of a recently detected n-wall room, it will be augmented with the semantic label stored in the database.

Fig. 8.8 shows the overall architecture of vS-Graphs with the fiducial marker detection and integration module, with recognized markers being stored in the *ATLAS Map Manager*. In brief, RGB-D data is processed in real time, providing integrated visual and depth information to subsequent modules. Meanwhile, *Fiducial Marker Detection* (utilizing the ArUco library in this work) operates independently on input frames, identifying markers *only if they are present* in the environment and associating them with the corresponding frames. Visual features are extracted and tracked across sequential frames in the *Tracking* thread, where pose information is either initialized or refined, depending on the map reconstruction stage, creating a 3D map with tracked features across frames. KeyFrame selection, a critical step following feature extraction, is performed within the *Tracking* thread by analyzing the visual data. These KeyFrames contain 3D map points, point clouds, and potentially detected fiducial markers, forming the foundation for subsequent processes.

Accordingly, each fiducial marker $\mathbf{m}_i^K = \{\mathbf{p}_m, \nu_m\}$ in G_t is constrained by the KeyFrame K observing it, where $\mathbf{p}_m \in SE(3)$ is the marker's pose and ν_m is its center point. The mentioned constraint is defined below:

$$\sigma_m({}^G\mathbf{K}, \mathbf{m}_i^K) = \| {}^L\mathbf{m}_i \boxplus {}^G\mathbf{K} \boxminus {}^G\mathbf{m}_{i_1} \|^2_{\mathbf{\Lambda}_{\tilde{\mathbf{m}}_i}} \tag{8.17}$$

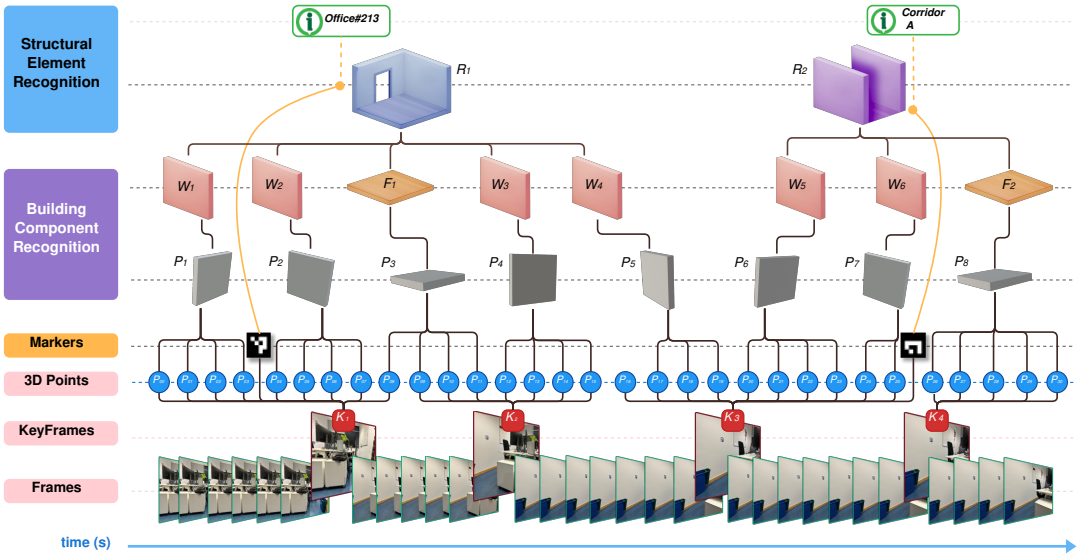


Figure 8.9: Scene graph structure generated using vS-Graphs, creating a hierarchical representation of the environment.

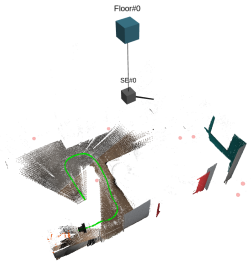
where ${}^{L}\mathbf{m}_i$ refers to the locally observed marker's pose, $\boxplus$ and $\boxminus$ represent the composition and inverse composition, $\|\dots\|$ is the *Mahalanobis* distance, and $\mathbf{\Lambda}_{\tilde{\mathbf{m}}_i}$ is marker's information matrix.

The marker $\mathbf{m}_i^{K_j} \in \mathbf{M}$ is associated with a structural element if it lies within the spatial bounds of the detected room/corridor $\delta^{K_g} \in \mathbf{\Delta}_\phi$. Thus, the Euclidean distance between the center point ν_m of $\mathbf{m}_i^{K_j}$ and the room/corridor's centroid ν_s, expressed as $d(\nu_m, \nu_s) \leq \epsilon_s$, where ϵ_s is the proximity threshold. The supporting condition is that ν_m must be enclosed among all bounding walls $\mathbf{\Pi}_{wall}$ with the normal vector $\mathbf{n}$:

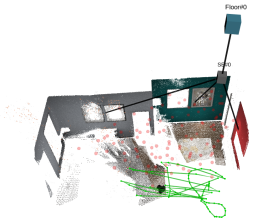
$$\forall \pi_{wall}^{K_i} \in \mathbf{\Pi}_{wall}, \quad (\mathbf{n}_{wall}^{K_i} \cdot (\nu_m - \nu_r)) \leq 0 \tag{8.18}$$

With this, Fig. 8.9 illustrates the extended scene graph structure of the vS-Graphs framework, emphasizing the role of fiducial markers in labeling the detected structural elements.

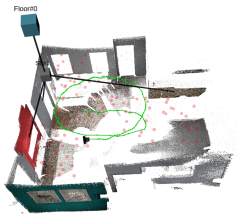
Appendix III. Generated Scene Graphs (Qualitative Analysis)



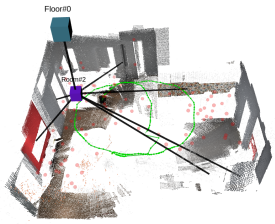
(a) ICL - deer-gr.



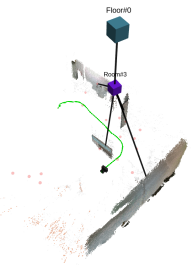
(b) ICL - deer-mavf.



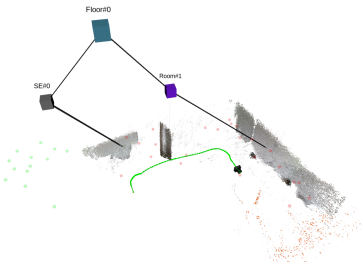
(c) ICL - deer-r.



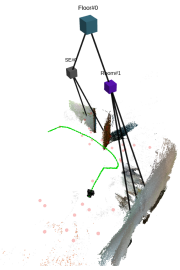
(d) ICL - deer-w.



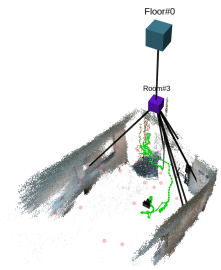
(e) OpenLoris - office-1-2.



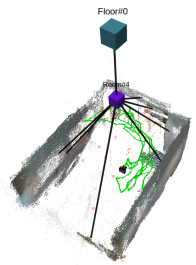
(f) OpenLoris - office-1-4.



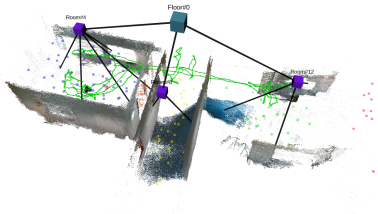
(g) OpenLoris - office-1-7.



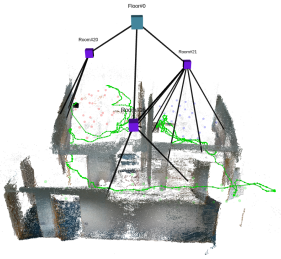
(h) AutoSense - SR02.



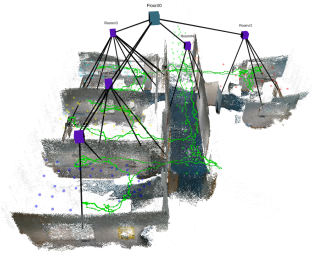
(i) AutoSense - SR03.



(j) AutoSense - MR01.



(k) AutoSense - MR02.



(l) AutoSense - MR03.

Figure 8.10: Scene graph examples generated by vS-Graphs across different environments, demonstrating consistent hierarchical semantic representation of structural components.

Part III

Appendices

9

Acronyms

- **6DoF**: **6**-**D**egrees-**o**f-**F**reedom
- **AOI**: **A**ngle **O**f **I**ncidence
- **AR**: **A**ugmented **R**eality
- **ATE**: **A**bsolute **T**rajectory **E**rror
- **BA**: **B**undle **A**djustment
- **BoW**: **B**ag of **W**ords
- **BRIEF**: **B**inary **R**obust **I**ndependent **E**lementary **F**eatures
- **CCA**: **C**onnected **C**omponent **A**nalysis
- **CLC**: **C**holesteric **L**iquid **C**rystal
- **CNN**: **C**onvolutional **N**eural **N**etwork
- **CRC**: **C**yclic **R**edundancy **C**heck
- **CSR**: **C**holesteric **S**pherical **R**eflector
- **CV**: **C**omputer **V**ision
- **DoF**: **D**egree **o**f **F**reedom
- **DNN**: **D**eep **N**eural **N**etwork
- **DTAM**: **D**ense **T**racking **A**nd **M**apping
- **EKF**: **E**xtended **K**alman **F**ilter
- **FAST**: **F**eatures from **A**ccelerated **S**egment **T**est

- **FoV**: **F**ield **o**f **V**iew
- **FPS**: **F**rames **P**er **S**econd
- **GBA**: **G**lobal **B**undle **A**djustment
- **GMS**: **G**rid-based **M**otion **S**tatistics
- **GNN**: **G**raph **N**eural **N**etwork
- **GPS**: **G**lobal **P**ositioning **S**ystem
- **GPU**: **G**raphics **P**rocessing **U**nit
- **GUI**: **G**raphical **U**ser **I**nterface
- **HOG**: **H**istogram of **O**riented **G**radients
- **HRI**: **H**uman–**R**obot **I**nteraction
- **IMU**: **I**nertial **M**easurement **U**nit
- **IR**: **I**nfra**r**ed
- **KLT**: **K**anade-**L**ucas-**T**omasi
- **LBA**: **L**ocal **B**undle **A**djustment
- **LC**: **L**iquid **C**rystal
- **LiDAR**: **Li**ght **D**etection **A**nd **R**anging
- **LSTM**: **L**ong **S**hort-**T**erm **M**emory
- **Mask R-CNN**: **M**ask **R**egion-based **CNN**
- **MAV**: **M**icro **A**erial **V**ehicle
- **MR**: **M**ixed **R**eality
- **NGO**: **N**eural **G**raph **O**ptimization
- **NIR**: **N**ear-**I**nfra**r**ed
- **NRCC**: **N**on-**R**igid **C**ontext **C**ulling
- **OGM**: **O**ccupancy **G**rid **M**apping
- **ORB**: **O**riented FAST and **R**otated BRIEF
- **PnP**: **P**erspective-**n**-**P**oint
- **PTAM**: **P**arallel **T**racking **A**nd **M**apping
- **RANSAC**: **R**andom **S**ample **C**onsensus
- **RCNN**: **R**ecurrent **C**onvolutional **N**eural **N**etwork

- **RGB-D**: **R**ed **G**reen **B**lue **D**epth
- **ROI**: **R**egion **O**f **I**nterest
- **ROS**: **R**obot **O**perating **S**ystem
- **RMSE**: **R**oot **M**ean **S**quare **D**eviation
- **RNN**: **R**ecurrent **N**eural **N**etwork
- **SIFT**: **S**cale-**I**nvariant **F**eature **T**ransform
- **SLAM**: **S**imultaneous **L**ocalization **A**nd **M**apping
- **STD**: **ST**andard **D**eviation
- **SURF**: **S**peeded-**U**p **R**obust **F**eatures
- **UAV**: **U**nmanned **A**erial **V**ehicle
- **UGV**: **U**nmanned **G**round **V**ehicle
- **UV**: **U**ltra**v**iolet
- **VGG**: **V**isual **G**eometry **G**roup
- **VO**: **V**isual **O**dometry
- **VSLAM**: **V**isual **SLAM**

10

Author's Research Contributions

This chapter lists all peer-reviewed **authored** or **co-authored** publications during the Ph.D. period. Publications that directly contribute to the content of this thesis are marked with a "▲" symbol, while the remaining publications represent broader research contributions. For additional context regarding publication quality, journal quartile rankings ($Q1$–$Q4$) from **Scopus**[1] and conference rankings (A^*, A, B, C) by **CORE2023**[2] are reported in **violet**.

1. H. Agha, Y. Geng, X. Ma, D.I. Avşar, R. Kizhakidathazhath, Y. Zhang, **A. Tourani**, H. Bavle, J.L. Sanchez-Lopez, H. Voos, M. Schwartz, and J. P.F. Lagerwall, "**Unclonable Human-invisible Machine Vision Markers Leveraging the Omnidirectional Chiral Bragg Diffraction of Cholesteric Spherical Reflectors**," *Light: Science & Applications (Nature)*, vol. 11, no. 1, p. 309, 2022. (`https://doi.org/10.1038/s41377-022-01002-4`, **Q1 (top-2%)**)

2. ▲ **A. Tourani**, H. Bavle, J.L. Sanchez-Lopez, and H. Voos, "**Visual SLAM: What are the Current Trends and What to Expect?**," *Sensors*, vol. 22, no. 23, p. 9297, 2022. (`https://doi.org/10.3390/s22239297`, **Q1 (top-12%)**)

3. **A. Tourani**, H. Bavle, J.L. Sanchez-Lopez, R. Muñoz Salinas, and H. Voos, "**Hierarchical Visual SLAM based on Fiducial Markers**," *International Conference on Robotics and Automation (ICRA'23) - Workshop on Active Methods in Autonomous Navigation*, London, UK, 2023. (**Rank-A***)

4. H. Bavle, J.L. Sanchez-Lopez, C. Cimarelli, **A. Tourani**, and H. Voos, "**From SLAM to Situational Awareness: Challenges and Survey**," *Sensors*, vol. 23, no. 10, p. 4849, 2023. (`https://doi.org/10.3390/s23104849`, **Q1 (top-12%)**)

5. ▲ **A. Tourani**, H. Bavle, J.L. Sanchez-Lopez, and H. Voos, "**Marker-based Visual SLAM leveraging Hierarchical Representations**," *Proceedings of the*

[1] `https://www.elsevier.com/products/scopus`
[2] `https://portal.core.edu.au/conf-ranks/`

IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'23), pp. 3461-3467, Detroit, MI, USA, 2023. (https://doi.org/10.1109/IROS55552.2023.10341891, **Rank-A**)

6. A. Radwan, **A. Tourani**, H. Bavle, H. Voos, and J.L. Sanchez-Lopez, "**UAV-assisted Visual SLAM Generating Reconstructed 3D Scene Graphs in GPS-denied Environments**," *Proceedings of the IEEE International Conference on Unmanned Aircraft Systems (ICUAS'24)*, pp. 1109-1116, Chania - Crete, Greece, 2024. (https://doi.org/10.1109/ICUAS60882.2024.10556948, **Rank-B**)

7. ▲ **A. Tourani**, H. Bavle, J.L. Sanchez-Lopez, D.I. Avşar, R. Muñoz Salinas, and H. Voos, "**Vision-Based Situational Graphs Exploiting Fiducial Markers for the Integration of Semantic Entities**," *Robotics*, vol. 13, no. 7, p. 106, 2024. (https://doi.org/10.3390/robotics13070106, **Q1 (top-18%)**)

8. **A. Tourani**, S. Ejaz, H. Bavle, J.L. Sanchez-Lopez, and H. Voos, "**Towards Localizing Structural Elements: Merging Geometrical Detection with Semantic Verification in RGB-D Data**," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'24) - 14th Workshop on Planning, Perception and Navigation for Intelligent Vehicles*, 2024. (https://arxiv.org/abs/2409.06625, **Rank-A**)

9. I.M. Chekam, I. Pastor-Martinez, **A. Tourani**, J.A. Millan-Romera, L. Ribeiro, P.M. Bastos Soares, H. Voos, and J.L. Sanchez-Lopez, "**Interpretable Robot Control via Structured Behavior Trees and Large Language Models,**" *IEEE Access*, vol. 13, pp. 200905-200916, 2025. (https://doi.org/10.1109/ACCESS.2025.3635471, **Q1 (top-10%)**)

10. S. Ejaz, M. Giberna, M. Shaheer, J.A. Millan-Romera, **A. Tourani**, P. Kremer, H. Voos, and J.L. Sanchez-Lopez, "**Situationally-aware Path Planning Exploiting 3D Scene Graphs,**" *IEEE Robotics and Automation Letters (RA-L)*, vol. 11, no. 3, pp. 3358-3365, 2026. (https://doi.org/10.1109/LRA.2026.3656775, **Q1 (top-3%)**)

11. P.M. Bastos Soares, **A. Tourani**, M. Fernandez-Cortizas, A. Bikandi Noya, H. Voos, and J.L. Sanchez-Lopez, "**SMapper: A Multi-Modal Data Acquisition Platform for SLAM Benchmarking**" *Journal of Intelligent & Robotic Systems (JINT)* vol. 112, no. 20, 2026. (https://doi.org/10.1007/s10846-026-02351-7, **Q1 (top-14%)**)

12. ▲ **A. Tourani**, D.I. Avşar, H. Bavle, J.L. Sanchez-Lopez, and H. Voos, "**Unveiling the Potential of iMarkers: Invisible Fiducial Markers for Advanced Robotics,**" *IEEE Robotics and Automation Magazine (RA-M - Conditionally Accepted)*, 2026. (https://doi.org/10.48550/arXiv.2501.15505, **Q1 (top-16%)**)

13. ▲ **A. Tourani**, S. Ejaz, H. Bavle, M. Fernandez-Cortizas, D. Morilla-Cabello, J.L. Sanchez-Lopez, and H. Voos "**vS-Graphs: Tightly Coupling Visual SLAM and 3D Scene Graphs Exploiting Hierarchical Scene Understanding,**" *IEEE Robotics and Automation Letters (RA-L - Under Review)*, 2026. (https://doi.org/10.48550/arXiv.2503.01783, **Q1 (top-3%)**)

14. L. Ribeiro, M. Shaheer, M. Fernandez-Cortizas, **A. Tourani**, H. Voos, and J.L. Sanchez-Lopez, "**Human Interaction for Collaborative Semantic SLAM using Extended Reality,**" *ACM Transactions on Computer-Human Interaction (Under Review)*, 2026. (`https://doi.org/10.48550/arXiv.2509.14949`, **Q1 (top-10%)**)

15. A. Bikandi-Noya, M. Fernandez-Cortizas, M. Shaheer, **A. Tourani**, H. Voos, and J.L. Sanchez-Lopez, "**BIM Informed Visual SLAM for Construction Monitoring,**" *IEEE Robotics and Automation Letters (RA-L - Under Review)*, 2026. (`https://doi.org/10.48550/arXiv.2509.13972`, **Q1 (top-3%)**)

11

Open-Source Software Contributions

The thesis includes a collection of **open-source projects** designed or contributed by the author to support reproducibility, help in community contributions, and enable further research and development. These resources are summarized below:

1. **vS-Graphs** framework [261, 168] (GPL-3.0 License), available on GitHub: `https://github.com/snt-arg/visual_sgraphs`.
2. **iMarker Detector (Standalone version)** framework [165, 164] (SnT Academic License), available on GitHub: `https://github.com/snt-arg/iMarker_detector_standalone`.
3. **iMarker Detector (ROS-enabled version)** framework [262, 164] (SnT Academic License), available on GitHub: `https://github.com/snt-arg/iMarker_detector_ros`.
4. **Semantic Scene Segmentation** framework [263] (GPL-3.0 License), available on GitHub: `https://github.com/snt-arg/scene_segment_ros`.
5. **Robot Suite** framework [264, 265] (GPL-3.0 License), available on GitHub: `https://github.com/snt-arg/robot_suite`.
6. **SMapper** framework [266, 267] (GPL-3.0 License), available on GitHub: `https://github.com/snt-arg/smapper`.

Bibliography

[1] D. Sliney, "What is light? the visible spectrum and beyond," *Eye*, vol. 30, no. 2, pp. 222–229, 2016.

[2] B. D. Guenther, *Modern optics*. Oxford University Press, 2018.

[3] R. Chipman, W. S. T. Lam, and G. Young, *Polarized light and optical systems*. CRC press, 2018.

[4] L. B. Wolff, "Polarization camera for computer vision with a beam splitter," *Journal of the Optical Society of America A*, vol. 11, no. 11, pp. 2935–2945, 1994.

[5] L. Koshin Kogaku Co., "An introduction to beamsplitters," https://www.koshin-kogaku.com/application-notes/an-introduction-to-beam-splitter, accessed: November 11, 2025.

[6] E. O. Inc., "What are beamsplitters?" https://www.edmundoptics.com/knowledge-center/application-notes/optics/what-are-beamsplitters, accessed: November 11, 2025.

[7] M. Schadt, "Liquid crystal materials and liquid crystal displays," *Annual review of materials science*, vol. 27, no. 1, pp. 305–379, 1997.

[8] M. Schwartz, G. Lenzini, Y. Geng, P. B. Rønne, P. Y. Ryan, and J. P. Lagerwall, "Cholesteric liquid crystal shells as enabling material for information-rich design and architecture," *Advanced Materials*, vol. 30, no. 30, p. 1707382, 2018.

[9] Y. Geng, J. Noh, I. Drevensek-Olenik, R. Rupp, G. Lenzini, and J. P. Lagerwall, "High-fidelity spherical cholesteric liquid crystal bragg reflectors generating unclonable patterns for secure authentication," *Scientific reports*, vol. 6, no. 1, p. 26840, 2016.

[10] M. Schwartz, Y. Geng, H. Agha, R. Kizhakidathazhath, D. Liu, G. Lenzini, and J. P. Lagerwall, "Linking physical objects to their digital twins via fiducial markers designed for invisibility to humans," *Multifunctional Materials*, vol. 4, no. 2, p. 022002, 2021.

[11] Y. Geng, R. Kizhakidathazhath, and J. P. Lagerwall, "Encoding hidden information onto surfaces using polymerized cholesteric spherical reflectors," *Advanced functional materials*, vol. 31, no. 21, p. 2100399, 2021.

[12] H. Agha, Y. Geng, X. Ma, D. I. Avşar, R. Kizhakidathazhath, Y.-S. Zhang, A. Tourani, H. Bavle, J.-L. Sanchez-Lopez, H. Voos *et al.*, "Unclonable human-invisible machine vision markers leveraging the omnidirectional chiral bragg diffraction of cholesteric spherical reflectors," *Light: Science & Applications*, vol. 11, no. 1, pp. 1–19, 2022.

[13] M. P. Arenas, H. Demirci, and G. Lenzini, "An analysis of cholesteric spherical reflector identifiers for object authenticity verification," *Machine Learning and Knowledge Extraction*, vol. 4, no. 1, pp. 222–239, 2022.

[14] M. Kalaitzakis, B. Cain, S. Carroll, A. Ambrosi, C. Whitehead, and N. Vitzilaios, "Fiducial markers for pose estimation: Overview, applications and experimental comparison of the artag, apriltag, aruco and stag markers," *Journal of Intelligent & Robotic Systems*, vol. 101, no. 4, p. 71, 2021.

[15] M. Kalaitzakis, S. Carroll, A. Ambrosi, C. Whitehead, and N. Vitzilaios, "Experimental comparison of fiducial markers for pose estimation," in *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2020, pp. 781–789.

[16] S. Tiwari, "An introduction to qr code technology," in *2016 International Conference on Information Technology (ICIT)*. IEEE, 2016, pp. 39–44.

[17] K. H. Pandya and H. J. Galiyawala, "A survey on qr codes: in context of research and application," *International Journal of Emerging Technology and Advanced Engineering*, vol. 4, no. 3, pp. 258–262, 2014.

[18] T. Korthals, D. Wolf, D. Rudolph, M. Hesse, and U. Rückert, "Fiducial marker-based extrinsic camera calibration for a robot benchmarking platform," in *2019 European Conference on Mobile Robots (ECMR)*. IEEE, 2019, pp. 1–6.

[19] X. X. Lu, "A review of solutions for perspective-n-point problem in camera pose estimation," in *Journal of Physics: Conference Series*, vol. 1087, no. 5. IOP Publishing, 2018, p. 052009.

[20] A. Aalerud, J. Dybedal, and G. Hovland, "Automatic calibration of an industrial rgb-d camera network using retroreflective fiducial markers," *Sensors*, vol. 19, no. 7, p. 1561, 2019.

[21] M. Hwang, B. Thananjeyan, S. Paradis, D. Seita, J. Ichnowski, D. Fer, T. Low, and K. Goldberg, "Efficiently calibrating cable-driven surgical robots with rgbd fiducial sensing and recurrent neural networks," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5937–5944, 2020.

[22] A. Reinke, M. Camurri, and C. Semini, "A factor graph approach to multi-camera extrinsic calibration on legged robots," in *2019 Third IEEE International Conference on Robotic Computing (IRC)*. IEEE, 2019, pp. 391–394.

[23] H. M. Balanji, A. E. Turgut, and L. T. Tunc, "A novel vision-based calibration framework for industrial robotic manipulators," *Robotics and Computer-Integrated Manufacturing*, vol. 73, p. 102248, 2022.

[24] E. Mráz, J. Rodina, and A. Babinec, "Using fiducial markers to improve localization of a drone," in *2020 23rd International Symposium on Measurement and Control in Robotics (ISMCR)*. IEEE, 2020, pp. 1–5.

[25] J. L. Sanchez-Lopez, V. Arellano-Quintana, M. Tognon, P. Campoy, and A. Franchi, "Visual marker-based multi-sensor fusion state estimation," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 16 003–16 008, 2017.

[26] S. M. Chacko and V. Kapila, "An augmented reality interface for human-robot interaction in unconstrained environments," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 3222–3228.

[27] X. Bian, W. Chen, D. Ran, Z. Liang, and X. Mei, "Fima-reader: A cost-effective fiducial marker reader system for autonomous mobile robot docking in manufacturing environments," *Applied Sciences*, vol. 13, no. 24, p. 13079, 2023.

[28] C. Elbrechter, R. Haschke, and H. Ritter, "Bi-manual robotic paper manipulation based on real-time marker tracking and physical modelling," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2011, pp. 1427–1432.

[29] N. Rogeau, V. Tiberghien, P. Latteur, and Y. Weinand, "Robotic insertion of timber joints using visual detection of fiducial markers," in *Proceedings of the 37th International Symposium on Automation and Robotics in Construction (ISARC)*, 2020, pp. 491–498.

[30] L. C. Klein, J. Braun, J. Mendes, V. H. Pinto, F. N. Martins, A. S. de Oliveira, H. Wörtche, P. Costa, and J. Lima, "A machine learning approach to robot localization using fiducial markers in robotatfactory 4.0 competition," *Sensors*, vol. 23, no. 6, p. 3128, 2023.

[31] U. Izagirre, I. Andonegui, L. Eciolaza, and U. Zurutuza, "Towards manufacturing robotics accuracy degradation assessment: A vision-based data-driven implementation," *Robotics and Computer-Integrated Manufacturing*, vol. 67, p. 102029, 2021.

[32] H. U. Odoemelem and K. Van Laerhoven, "A low-cost prototyping framework for human-robot desk interaction," in *Adjunct Proceedings of the 2020 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2020 ACM International Symposium on Wearable Computers*, 2020, pp. 191–194.

[33] A. Jain, M. Singhal, and M. Jhamb, "Aruco marker-based pick and place approach using a ur5 robotic arm and vacuum gripper," in *International Conference on Artificial Intelligence on Textile and Apparel*. Springer, 2023, pp. 365–379.

[34] L. Roveda, N. Castaman, S. Ghidoni, P. Franceschi, N. Boscolo, E. Pagello, and N. Pedrocchi, "Human-robot cooperative interaction control for the installation of heavy and bulky components," in *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2018, pp. 339–344.

[35] D. Khan, S. Ullah, D.-M. Yan, I. Rabbi, P. Richard, T. Hoang, M. Billinghurst, and X. Zhang, "Robust tracking through the design of high-quality fiducial markers: an optimization tool for artoolkit," *IEEE access*, vol. 6, pp. 22 421–22 433, 2018.

[36] F. J. Romero-Ramirez, R. Muñoz-Salinas, and R. Medina-Carnicer, "Tracking fiducial markers with discriminative correlation filters," *Image and Vision Computing*, vol. 107, p. 104094, 2021.

[37] H. Lim and Y. S. Lee, "Real-time single camera slam using fiducial markers," in *2009 ICCAS-SICE*. IEEE, 2009, pp. 177–182.

[38] B. Al-Tawil, T. Hempel, A. Abdelrahman, and A. Al-Hamadi, "A review of visual slam for robotics: Evolution, properties, and future applications," *Frontiers in Robotics and AI*, vol. 11, p. 1347985, 2024.

[39] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[40] A. Gupta and X. Fernando, "Simultaneous localization and mapping (slam) and data fusion in unmanned aerial vehicles: Recent advances and challenges," *Drones*, vol. 6, no. 4, p. 85, 2022.

[41] I. Vallivaara, J. Haverinen, A. Kemppainen, and J. Röning, "Magnetic field-based slam method for solving the localization problem in mobile robot floor-cleaning task," in *2011 15th international conference on advanced robotics (ICAR)*. IEEE, 2011, pp. 198–203.

[42] Q. Zou, Q. Sun, L. Chen, B. Nie, and Q. Li, "A comparative analysis of lidar slam-based indoor navigation for autonomous vehicles," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[43] D. Geromichalos, M. Azkarate, E. Tsardoulias, L. Gerdes, L. Petrou, and C. Perez Del Pulgar, "Slam for autonomous planetary rovers with global localization," *Journal of Field Robotics*, vol. 37, no. 5, pp. 830–847, 2020.

[44] T. Yang, P. Li, H. Zhang, J. Li, and Z. Li, "Monocular vision slam-based uav autonomous landing in emergencies and unknown environments," *Electronics*, vol. 7, no. 5, p. 73, 2018.

[45] Z. Liu, H. Chen, H. Di, Y. Tao, J. Gong, G. Xiong, and J. Qi, "Real-time 6d lidar slam in large-scale natural terrains for ugv," in *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 662–667.

[46] W. Chen, G. Shang, A. Ji, C. Zhou, X. Wang, C. Xu, Z. Li, and K. Hu, "An overview on visual slam: From tradition to semantic," *Remote Sensing*, vol. 14, no. 13, p. 3010, 2022.

[47] M. Filipenko and I. Afanasyev, "Comparison of various slam systems for mobile robot in an indoor environment," in *2018 International Conference on Intelligent Systems (IS)*. IEEE, 2018, pp. 400–407.

[48] G. Jia, X. Li, D. Zhang, W. Xu, H. Lv, Y. Shi, and M. Cai, "Visual-slam classical framework and key techniques: a review," *Sensors*, vol. 22, no. 12, p. 4582, 2022.

[49] K. Yousif, A. Bab-Hadiashar, and R. Hoseinnezhad, "An overview to visual odometry and visual slam: Applications to mobile robotics," *Intelligent Industrial Systems*, vol. 1, no. 4, pp. 289–311, 2015.

[50] B. Williams, M. Cummins, J. Neira, P. Newman, I. Reid, and J. Tardós, "A comparison of loop closing techniques in monocular slam," *Robotics and Autonomous Systems*, vol. 57, no. 12, pp. 1188–1197, 2009.

[51] A. Tourani, H. Bavle, J. L. Sanchez-Lopez, R. M. Salinas, and H. Voos, "Marker-based visual slam leveraging hierarchical representations," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023, pp. 3461–3467.

[52] A. Tourani, H. Bavle, D. I. Avşar, J. L. Sanchez-Lopez, R. Munoz-Salinas, and H. Voos, "Vision-based situational graphs exploiting fiducial markers for the integration of semantic entities," *Robotics*, vol. 13, no. 7, p. 106, 2024.

[53] L. Naimark and E. Foxlin, "Circular data matrix fiducial system and robust image processing for a wearable vision-inertial self-tracker," in *Proceedings. International Symposium on Mixed and Augmented Reality*, 2002, pp. 27–36.

[54] C. N. Klokmose, J. B. Kristensen, R. Bagge, and K. Halskov, "Bullseye: High-precision fiducial tracking for table-based tangible interaction," in *Proceedings of the Ninth ACM international conference on interactive tabletops and surfaces*, 2014, pp. 269–278.

[55] A. C. Rice, A. R. Beresford, and R. K. Harle, "Cantag: an open source software toolkit for designing and deploying marker-based vision systems," in *Fourth Annual IEEE International Conference on Pervasive Computing and Communications (PERCOM'06)*. IEEE, 2006, pp. 10–pp.

[56] L. Calvet, P. Gurdjos, and V. Charvillat, "Camera tracking using concentric circle markers: Paradigms and algorithms," in *2012 19th IEEE International Conference on Image Processing*. IEEE, 2012, pp. 1361–1364.

[57] P. Lightbody, T. Krajník, and M. Hanheide, "A versatile high-performance visual fiducial marker detection system with scalable identity encoding," in *Proceedings of the Symposium on Applied Computing*, 2017, pp. 276–282.

[58] F. Bergamasco, A. Albarelli, and A. Torsello, "Pi-tag: A fast image-space marker design based on projective invariants," *Machine Vision and Applications*, vol. 24, pp. 1295–1310, 2013.

[59] H. Uchiyama and H. Saito, "Random dot markers," in *2011 IEEE Virtual Reality Conference*. IEEE, 2011, pp. 35–38.

[60] F. Bergamasco, A. Albarelli, E. Rodola, and A. Torsello, "Rune-tag: A high accuracy fiducial marker with strong occlusion resilience," in *2011 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'11)*. IEEE, 2011, pp. 113–120.

[61] H. Nishino, "A 6dof fiducial tracking method based on topological region adjacency and angle information for tangible interaction," in *Proceedings of the Fourth International Conference on Tangible, Embedded, and Embodied Interaction*, 2010, pp. 253–256.

[62] ——, "A shape-free, designable 6-dof marker tracking method for camera-based interaction in mobile environment," in *Proceedings of the 18th ACM International Conference on Multimedia*, 2010, pp. 1055–1058.

[63] E. Costanza, S. B. Shelley, and J. Robinson, "D-touch: A consumer-grade tangible interface module and musical applications," in *Conference on Human-Computer Interaction (HCI03)*, 2003.

[64] H. Nishino, "Topolo surface: A 2d fiducial tracking system based on topological region adjacency and angle information," *Information and Media Technologies*, vol. 5, no. 2, pp. 479–488, 2010.

[65] R. Bencina, M. Kaltenbrunner, and S. Jorda, "Improved topological fiducial tracking in the reactivision system," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)-Workshops*. IEEE, 2005, pp. 99–99.

[66] A. Bookstein, V. A. Kulyukin, and T. Raita, "Generalized hamming distance," *Information Retrieval*, vol. 5, no. 4, pp. 353–375, 2002.

[67] J. S. Sobolewski, "Cyclic redundancy check," in *Encyclopedia of Computer Science*, 2003, pp. 476–479.

[68] M. Rohs, "Real-world interaction with camera phones," in *International Symposium on Ubiquitous Computing Systems*. Springer, 2004, pp. 74–89.

[69] J. Rekimoto and Y. Ayatsuka, "Cybercode: designing augmented reality environments with visual tags," in *Proceedings of DARE 2000 on Designing Augmented Reality environments*, 2000, pp. 1–10.

[70] V. Engine, "Vumarks," https://developer.vuforia.com/library/objects/vumarks, accessed: November 08, 2024.

[71] G. Yu, Y. Hu, and J. Dai, "TopoTag: A Robust and Scalable Topological Fiducial Marker System," *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, vol. 27, no. 9, pp. 3769–3780, 2021.

[72] B. Benligiray, C. Topal, and C. Akinlar, "Stag: A stable fiducial marker system," *Image and Vision Computing*, vol. 89, pp. 158–169, 2019.

[73] J. Rekimoto, "Matrix: A real-time object identification and registration method for augmented reality," in *Proceedings. 3rd Asia Pacific Computer-Human Interaction (Cat. No. 98EX110)*. IEEE, 1998, pp. 63–68.

[74] J. DeGol, T. Bretl, and D. Hoiem, "Chromatag: A colored marker and fast detection algorithm," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1472–1481.

[75] H. Kato and M. Billinghurst, "Marker tracking and hmd calibration for a video-based augmented reality conferencing system," in *Proceedings 2nd IEEE and ACM International Workshop on Augmented Reality (IWAR'99)*. IEEE, 1999, pp. 85–94.

[76] D. Wagner and D. Schmalstieg, "Artoolkitplus for pose tracking on mobile devices," 2007.

[77] M. Fiala, "Artag, a fiducial marker system using digital techniques," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 2. IEEE, 2005, pp. 590–596.

[78] B. Atcheson, F. Heide, and W. Heidrich, "Caltag: High precision fiducial markers for camera calibration." in *Vision, Modeling, and Visualization*, vol. 10, 2010, pp. 41–48.

[79] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[80] D. Flohr and J. Fischer, "A lightweight id-based extension for marker tracking systems," in *IPT-EGVE Symposium*, 2007.

[81] E. Olson, "Apriltag: A robust and flexible visual fiducial system," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 3400–3407.

[82] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, 2014.

[83] F. J. Romero-Ramire, R. Munoz-Salinas, and R. Medina-Carnicer, "Fractal markers: a new approach for long-range marker pose estimation under occlusion," *IEEE Access*, vol. 7, pp. 169 908–169 919, 2019.

[84] Z. Zhang, Y. Hu, G. Yu, and J. Dai, "Deeptag: a general framework for fiducial marker design and detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 3, pp. 2931–2944, 2022.

[85] C. Scheirer and C. Harrison, "Dynatags: Low-cost fiducial marker mechanisms," in *Proceedings of the 2022 International Conference on Multimodal Interaction*, 2022, pp. 432–443.

[86] J.-K. Huang, S. Wang, M. Ghaffari, and J. W. Grizzle, "Lidartag: A real-time fiducial tag system for point clouds," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4875–4882, 2021.

[87] F. Digiacomo, F. Bologna, F. Inglese, C. Stefanini, and M. Milazzo, "Mechatag: A mechanical fiducial marker and the detection algorithm," *Journal of Intelligent & Robotic Systems*, vol. 103, no. 3, p. 46, 2021.

[88] C. Getschmann and F. Echtler, "Seedmarkers: Embeddable markers for physical objects," in *Proceedings of the Fifteenth International Conference on Tangible, Embedded, and Embodied Interaction*, 2021, pp. 1–11.

[89] Y. Xie, L. Deng, T. Sun, Y. Fu, J. Li, X. Cui, H. Yin, S. Deng, J. Xiao, and B. Chen, "A4lidartag: Depth-based fiducial marker for extrinsic calibration of solid-state lidar and camera," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 6487–6494, 2022.

[90] Y. Wang, Y. Ji, D. Liu, Y. Tamura, H. Tsuchiya, A. Yamashita, and H. Asama, "Acmarker: Acoustic camera-based fiducial marker system in underwater environment," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5018–5025, 2020.

[91] H. Uchiyama and Y. Oyamada, "Transparent random dot markers," in *2018 24th International Conference on Pattern Recognition (ICPR)*. IEEE, 2018, pp. 254–259.

[92] M. D. Dogan, R. Garcia-Martin, P. W. Haertel, J. J. O'Keefe, A. Taka, A. Aurora, R. Sanchez-Reillo, and S. Mueller, "Brightmarker: 3d printed fluorescent markers for object tracking," in *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, 2023, pp. 1–13.

[93] M. D. Dogan, A. Taka, M. Lu, Y. Zhu, A. Kumar, A. Gupta, and S. Mueller, "Infraredtags: Embedding invisible ar markers and barcodes using low-cost, infrared-based 3d printing and imaging tools," in *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, 2022, pp. 1–12.

[94] K. D. Willis and A. D. Wilson, "Infrastructs: fabricating information inside physical objects for imaging in the terahertz region," *ACM Transactions on Graphics (TOG)*, vol. 32, no. 4, pp. 1–10, 2013.

[95] D. Li, A. S. Nair, S. K. Nayar, and C. Zheng, "Aircode: Unobtrusive physical tags for digital fabrication," in *Proceedings of the 30th annual ACM symposium on user interface software and technology*, 2017, pp. 449–460.

[96] C. Duan, S. Junginger, J. Huang, K. Jin, and K. Thurow, "Deep Learning for Visual SLAM in Transportation Robotics: A Review," *Transportation Safety and Environment*, vol. 1, no. 3, pp. 177–184, 01 2020. [Online]. Available: https://doi.org/10.1093/tse/tdz019

[97] M. Outahar, G. Moreau, and J.-M. Normand, "Direct and indirect vslam fusion for augmented reality," *Journal of Imaging*, vol. 7, no. 8, p. 141, 2021.

[98] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, 2007.

[99] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *2007 6th IEEE and ACM international symposium on mixed and augmented reality*. IEEE, 2007, pp. 225–234.

[100] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "Dtam: Dense tracking and mapping in real-time," in *2011 international conference on computer vision*. IEEE, 2011, pp. 2320–2327.

[101] D. Zou and P. Tan, "Coslam: Collaborative visual slam in dynamic environments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 2, pp. 354–366, 2012.

[102] S. Urban and S. Hinz, "Multicol-slam-a modular real-time multi-camera slam system," *arXiv preprint arXiv:1610.07336*, 2016.

[103] Y. Zhu, C. Zheng, C. Yuan, X. Huang, and X. Hong, "Camvox: A low-cost and accurate lidar-assisted visual slam system," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 5049–5055.

[104] T.-M. Nguyen, S. Yuan, M. Cao, T. H. Nguyen, and L. Xie, "Viral slam: Tightly coupled camera-imu-uwb-lidar slam," *arXiv preprint arXiv:2105.03296*, 2021.

[105] A. R. Vidal, H. Rebecq, T. Horstschaefer, and D. Scaramuzza, "Ultimate slam? combining events, images, and imu for robust visual slam in hdr and high-speed scenarios," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 994–1001, 2018.

[106] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: a versatile and accurate monocular slam system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[107] R. Mur-Artal and J. D. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[108] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.

[109] A. Pumarola, A. Vakhitov, A. Agudo, A. Sanfeliu, and F. Moreno-Noguer, "Pl-slam: Real-time monocular visual slam with points and lines," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 4503–4508.

[110] R. Gomez-Ojeda, F.-A. Moreno, D. Zuniga-Noël, D. Scaramuzza, and J. Gonzalez-Jimenez, "Pl-slam: A stereo slam system through the combination of points and line segments," *IEEE Transactions on Robotics*, vol. 35, no. 3, pp. 734–746, 2019.

[111] H. Zhou, D. Zou, L. Pei, R. Ying, P. Liu, and W. Yu, "Structslam: Visual slam with building structure lines," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 4, pp. 1364–1375, 2015.

[112] H. Lim, Y. Kim, K. Jung, S. Hu, and H. Myung, "Avoiding degeneracy for monocular visual slam with point and line features," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 11 675–11 681.

[113] S. Bultmann, K. Li, and U. D. Hanebeck, "Stereo visual slam based on unscented dual quaternion filtering," in *2019 22th International Conference on Information Fusion (FUSION)*. IEEE, 2019, pp. 1–8.

[114] R. Munoz-Salinas, M. J. Marin-Jimenez, and R. Medina-Carnicer, "Spm-slam: Simultaneous localization and mapping with squared planar markers," *Pattern Recognition*, vol. 86, pp. 156–171, 2019.

[115] M. Bloesch, J. Czarnowski, R. Clark, S. Leutenegger, and A. J. Davison, "Codeslam—learning a compact, optimisable representation for dense visual slam," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2560–2568.

[116] B. Li, D. Zou, D. Sartori, L. Pei, and W. Yu, "Textslam: Visual slam with planar text features," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 2102–2108.

[117] H. M. S. Bruno and E. L. Colombini, "Lift-slam: A deep-learning feature-based monocular visual slam method," *Neurocomputing*, vol. 455, pp. 97–110, 2021.

[118] K. Naveed, M. L. Anjum, W. Hussain, and D. Lee, "Deep introspective slam: deep reinforcement learning based approach to avoid tracking failure in visual slam," *Autonomous Robots*, vol. 46, no. 6, pp. 705–724, 2022.

[119] Q. Peng, Z. Xiang, Y. Fan, T. Zhao, and X. Zhao, "Rwt-slam: Robust visual slam for highly weak-textured environments," *arXiv preprint arXiv:2207.03539*, 2022.

[120] Y. Sun, J. Hu, J. Yun, Y. Liu, D. Bai, X. Liu, G. Zhao, G. Jiang, J. Kong, and B. Chen, "Multi-objective location and mapping based on deep learning and visual slam," *Sensors*, vol. 22, no. 19, p. 7576, 2022.

[121] D. Li, X. Shi, Q. Long, S. Liu, W. Yang, F. Wang, Q. Wei, and F. Qiao, "Dxslam: A robust and efficient visual slam system with deep features," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 4958–4965.

[122] G. Li, L. Yu, and S. Fei, "A deep-learning real-time visual slam system based on multi-task feature extraction network and self-supervised feature points," *Measurement*, vol. 168, p. 108403, 2021.

[123] A. Steenbeek and F. Nex, "Cnn-based dense monocular visual slam for real-time uav exploration in emergency conditions," *Drones*, vol. 6, no. 3, p. 79, 2022.

[124] P. Su, S. Luo, and X. Huang, "Real-time dynamic slam algorithm based on deep learning," *IEEE Access*, vol. 10, pp. 87 754–87 766, 2022.

[125] J. Chen, F. Xie, L. Huang, J. Yang, X. Liu, and J. Shi, "A robot pose estimation optimized visual slam algorithm based on co-hdc instance segmentation network for dynamic scenes," *Remote Sensing*, vol. 14, no. 9, p. 2114, 2022.

[126] X.-Y. Dai, Q.-H. Meng, W.-J. Zheng, and S.-K. Zhu, "Monocular visual slam based on vgg feature point extraction," in *2020 39th Chinese Control Conference (CCC)*. IEEE, 2020, pp. 3375–3380.

[127] C. Yu, Z. Liu, X.-J. Liu, F. Xie, Y. Yang, Q. Wei, and Q. Fei, "Ds-slam: A semantic visual slam towards dynamic environments," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1168–1174.

[128] L. Cui and C. Ma, "Sof-slam: A semantic visual slam for dynamic environments," *IEEE access*, vol. 7, pp. 166 528–166 539, 2019.

[129] B. Bescos, J. M. Fácil, J. Civera, and J. Neira, "Dynaslam: Tracking, mapping, and inpainting in dynamic scenes," *IEEE robotics and automation letters*, vol. 3, no. 4, pp. 4076–4083, 2018.

[130] B. Bescos, C. Campos, J. D. Tardós, and J. Neira, "Dynaslam ii: Tightly-coupled multi-object tracking and slam," *IEEE robotics and automation letters*, vol. 6, no. 3, pp. 5191–5198, 2021.

[131] L. Xiao, J. Wang, X. Qiu, Z. Rong, and X. Zou, "Dynamic-slam: Semantic monocular visual localization and mapping based on deep learning in dynamic environment," *Robotics and Autonomous Systems*, vol. 117, pp. 1–16, 2019.

[132] S. Wen, X. Li, X. Liu, J. Li, S. Tao, Y. Long, and T. Qiu, "Dynamic slam: A visual slam in outdoor dynamic scenes," *IEEE Transactions on Instrumentation and Measurement*, 2023.

[133] X. Yuan and S. Chen, "Sad-slam: A visual slam based on semantic and depth information," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 4930–4935.

[134] J. Cheng, Y. Sun, and M. Q.-H. Meng, "Improving monocular visual slam in dynamic environments: an optical-flow-based approach," *Advanced Robotics*, vol. 33, no. 12, pp. 576–589, 2019.

[135] S. Yang, C. Zhao, Z. Wu, Y. Wang, G. Wang, and D. Li, "Visual slam based on semantic segmentation and geometric constraints for dynamic indoor environments," *IEEE Access*, vol. 10, pp. 69 636–69 649, 2022.

[136] S. L. Bowman, N. Atanasov, K. Daniilidis, and G. J. Pappas, "Probabilistic data association for semantic slam," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 1722–1729.

[137] K. Doherty, D. Baxter, E. Schneeweiss, and J. Leonard, "Probabilistic data association via mixture models for robust semantic slam," 2019.

[138] T. Whelan, S. Leutenegger, R. F. Salas-Moreno, B. Glocker, and A. J. Davison, "Elasticfusion: Dense slam without a pose graph." in *Robotics: science and systems*, vol. 11. Rome, Italy, 2015, p. 3.

[139] C. Yan, D. Qu, D. Xu, B. Zhao, Z. Wang, D. Wang, and X. Li, "Gs-slam: Dense visual slam with 3d gaussian splatting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 19 595–19 604.

[140] Z. Peng, T. Shao, Y. Liu, J. Zhou, Y. Yang, J. Wang, and K. Zhou, "Rtg-slam: Real-time 3d reconstruction at scale using gaussian splatting," in *ACM SIGGRAPH 2024 Conference Papers*, 2024, pp. 1–11.

[141] N. Keetha, J. Karhade, K. M. Jatavallabhula, G. Yang, S. Scherer, D. Ramanan, and J. Luiten, "Splatam: Splat track & map 3d gaussians for dense rgb-d slam," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 21 357–21 366.

[142] Z. Zhu, S. Peng, V. Larsson, W. Xu, H. Bao, Z. Cui, M. R. Oswald, and M. Pollefeys, "Nice-slam: Neural implicit scalable encoding for slam," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 12 786–12 796.

[143] J. Xu, H. Cao, D. Li, K. Huang, C. Qian, L. Shangguan, and Z. Yang, "Edge assisted mobile semantic visual slam," in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*. IEEE, 2020, pp. 1828–1837.

[144] A. J. Ben Ali, Z. S. Hashemifar, and K. Dantu, "Edge-slam: edge-assisted visual simultaneous localization and mapping," in *Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services*, 2020, pp. 325–337.

[145] D. Schlegel, M. Colosi, and G. Grisetti, "Proslam: Graph slam from a programmer's perspective," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 3833–3840.

[146] H. Bavle, P. De La Puente, J. P. How, and P. Campoy, "Vps-slam: Visual planar semantic slam for aerial robotic systems," *IEEE Access*, vol. 8, pp. 60 704–60 718, 2020.

[147] S. Sumikura, M. Shibuya, and K. Sakurada, "Openvslam: a versatile visual slam framework," in *Proceedings of the 27th ACM International Conference on Multimedia*, 2019, pp. 2292–2295.

[148] M. Ferrera, A. Eudes, J. Moras, M. Sanfourche, and G. Le Besnerais, "Ov^{2}slam: A fully online and versatile visual slam for real-time applications," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1399–1406, 2021.

[149] Z. Teed and J. Deng, "Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[150] R. Muñoz-Salinas and R. Medina-Carnicer, "Ucoslam: Simultaneous localization and mapping by fusion of keypoints and squared planar markers," *Pattern Recognition*, vol. 101, p. 107193, 2020.

[151] F. J. Romero-Ramirez, R. Muñoz-Salinas, M. J. Marín-Jiménez, M. Cazorla, and R. Medina-Carnicer, "sslam: Speeded-up visual slam mixing artificial markers and temporary keypoints," *Sensors*, vol. 23, no. 4, p. 2210, 2023.

[152] B. Pfrommer and K. Daniilidis, "Tagslam: Robust slam with fiducial markers," 2019. [Online]. Available: https://arxiv.org/abs/1910.00679

[153] K. An, H. Fan, and J. Dong, "A marker-based method for visual-inertial initialization," *Intelligent Marine Technology and Systems*, vol. 2, no. 1, p. 27, 2024.

[154] Z. Wang, Z. Zhang, W. Zhu, X. Hu, H. Deng, G. He, and X. Kang, "A robust planar marker-based visual slam," *Sensors*, vol. 23, no. 2, p. 917, 2023.

[155] H. Bavle, J. L. Sanchez-Lopez, M. Shaheer, J. Civera, and H. Voos, "Situational graphs for robot navigation in structured indoor environments," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 9107–9114, 2022.

[156] ——, "S-graphs+: Real-time localization and mapping leveraging hierarchical representations," *IEEE Robotics and Automation Letters*, vol. 8, no. 8, pp. 4927–4934, 2023.

[157] N. Hughes, Y. Chang, and L. Carlone, "Hydra: A real-time spatial perception system for 3d scene graph construction and optimization," 2022.

[158] A. Werby, C. Huang, M. Büchner, A. Valada, and W. Burgard, "Hierarchical open-vocabulary 3d scene graphs for language-grounded robot navigation," in *First Workshop on Vision-Language Models for Navigation and Manipulation at ICRA 2024*, 2024.

[159] I. Armeni, Z.-Y. He, J. Gwak, A. R. Zamir, M. Fischer, J. Malik, and S. Savarese, "3d scene graph: A structure for unified semantics, 3d space, and camera," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 5664–5673.

[160] S.-C. Wu, J. Wald, K. Tateno, N. Navab, and F. Tombari, "Scenegraphfusion: Incremental 3d scene graph prediction from rgb-d sequences," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 7515–7525.

[161] A. Rosinol, A. Gupta, M. Abate, J. Shi, and L. Carlone, "3d dynamic scene graphs: Actionable spatial perception with places, objects, and humans," 2020.

[162] H. Bavle, J. L. Sanchez-Lopez, M. Shaheer, J. Civera, and H. Voos, "S-graphs 2.0–a hierarchical-semantic optimization and loop closure for slam," *arXiv preprint arXiv:2502.18044*, 2025.

[163] H. Bavle, J. L. Sanchez-Lopez, C. Cimarelli, A. Tourani, and H. Voos, "From slam to situational awareness: Challenges and survey," *Sensors*, vol. 23, no. 10, p. 4849, 2023.

[164] A. Tourani, D. I. Avsar, H. Bavle, J. L. Sanchez-Lopez, J. Lagerwall, and H. Voos, "Unveiling the potential of imarkers: Invisible fiducial markers for advanced robotics," *arXiv preprint arXiv:2501.15505*, 2025.

[165] A. Tourani, D. I. Avsar, J. L. Sanchez-Lopez, J. Lagerwall, and H. Voos, "imarker detector (standalone version)," https://github.com/snt-arg/iMarker_detector_standalone, 2022.

[166] A. Tourani, H. Bavle, J. L. Sanchez-Lopez, and H. Voos, "Visual slam: What are the current trends and what to expect?" *Sensors*, vol. 22, no. 23, p. 9297, 2022.

[167] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *2012 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2012, pp. 573–580.

[168] A. Tourani, S. Ejaz, H. Bavle, M. Fernandez-Cortizas, D. Morilla-Cabello, J. L. Sanchez-Lopez, and H. Voos, "vs-graphs: Tightly coupling visual slam and 3d scene graphs exploiting hierarchical scene understanding," *arXiv preprint arXiv:2503.01783*, 2025.

[169] J. A. Millan-Romera, H. Bavle, M. Shaheer, M. R. Oswald, H. Voos, and J. L. Sanchez-Lopez, "Learning high-level semantic-relational concepts for slam," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2024, pp. 9803–9810.

[170] A. Bikandi-Noya, M. Fernandez-Cortizas, M. Shaheer, A. Tourani, H. Voos, and J. L. Sanchez-Lopez, "Bim informed visual slam for construction monitoring," *arXiv preprint arXiv:2509.13972*, 2025.

[171] M. Kalaitzakis, B. Cain, S. Carroll, A. Ambrosi, C. Whitehead, and N. Vitzilaios, "Fiducial markers for pose estimation: Overview, applications and experimental comparison of the artag, apriltag, aruco and stag markers," *Journal of Intelligent & Robotic Systems*, vol. 101, pp. 1–26, 2021.

[172] G. M. Costa, M. R. Petry, J. G. Martins, and A. P. Moreira, "Assessment of multiple fiducial marker trackers on hololens 2," *IEEE Access*, 2024.

[173] N. Ayala, D. Mardanbegi, A. Zafar, E. Niechwiej-Szwedo, S. Cao, S. Kearns, E. Irving, and A. T. Duchowski, "Does fiducial marker visibility impact task performance and information processing in novice and low-time pilots?" *Computers & Graphics*, vol. 119, p. 103889, 2024.

[174] R. M. Claro, D. B. Silva, and A. M. Pinto, "Artuga: A novel multimodal fiducial marker for aerial robotics," *Robotics and Autonomous Systems*, vol. 163, p. 104398, 2023.

[175] A. R. Khairuddin, M. S. Talib, and H. Haron, "Review on simultaneous localization and mapping (slam)," in *2015 IEEE International Conference on Control Systems, Computing and Engineering (ICCSCE)*. IEEE, 2015, pp. 85–90.

[176] J. Li, Y. Bi, M. Lan, H. Qin, M. Shan, F. Lin, and B. M. Chen, "Real-time simultaneous localization and mapping for uav: A survey," in *Proc. of International micro air vehicle competition and conference*, vol. 2016, 2016, p. 237.

[177] M. Zaffar, S. Ehsan, R. Stolkin, and K. M. Maier, "Sensors, slam and long-term autonomy: a review," in *2018 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*. IEEE, 2018, pp. 285–290.

[178] X. Gao and T. Zhang, *Introduction to Visual SLAM: From Theory to Practice*. Springer Nature, 2021.

[179] Y.-J. Yeh and H.-Y. Lin, "3d reconstruction and visual slam of indoor scenes for augmented reality application," in *2018 IEEE 14th International Conference on Control and Automation (ICCA)*. IEEE, 2018, pp. 94–99.

[180] A. Macario Barros, M. Michel, Y. Moline, G. Corre, and F. Carrel, "A comprehensive survey of visual slam algorithms," *Robotics*, vol. 11, no. 1, p. 24, 2022.

[181] Y. Jia, X. Yan, and Y. Xu, "A survey of simultaneous localization and mapping for robots," in *2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, vol. 1. IEEE, 2019, pp. 857–861.

[182] I. A. Kazerouni, L. Fitzgerald, G. Dooly, and D. Toal, "A survey of state-of-the-art on visual slam," *Expert Systems with Applications*, p. 117734, 2022.

[183] S. Arshad and G.-W. Kim, "Role of deep learning in loop closure detection for visual and lidar slam: A survey," *Sensors*, vol. 21, no. 4, p. 1243, 2021.

[184] A. Singandhupe and H. M. La, "A review of slam techniques and security in autonomous driving," in *2019 Third IEEE International Conference on Robotic Computing (IRC)*. IEEE, 2019, pp. 602–607.

[185] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE conference on computer vision and pattern recognition*. IEEE, 2012, pp. 3354–3361.

[186] M. R. U. Saputra, A. Markham, and N. Trigoni, "Visual slam and structure from motion in dynamic environments: A survey," *ACM Comput. Surv.*, vol. 51, no. 2, feb 2018. [Online]. Available: https://doi.org/10.1145/3177853

[187] M. I. Ribeiro, "Kalman and extended kalman filters: Concept, derivation and properties," *Institute for Systems and Robotics*, vol. 43, p. 46, 2004.

[188] D. G. Viswanathan, "Features from accelerated segment test (fast)," in *Proceedings of the 10th workshop on image analysis for multimedia interactive services, London, UK*, 2009, pp. 6–8.

[189] F. Endres, J. Hess, J. Sturm, D. Cremers, and W. Burgard, "3-d mapping with an rgb-d camera," *IEEE Transactions on Robotics*, vol. 30, no. 1, pp. 177–187, 2013.

[190] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison, "Slam++: Simultaneous localisation and mapping at the level of objects," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 1352–1359.

[191] E. Mendes, P. Koch, and S. Lacroix, "Icp-based pose-graph slam," in *2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. IEEE, 2016, pp. 195–200.

[192] M. He, C. Zhu, Q. Huang, B. Ren, and J. Liu, "A review of monocular visual odometry," *The Visual Computer*, vol. 36, no. 5, pp. 1053–1065, 2020.

[193] G. Gallego, T. Delbrück, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. J. Davison, J. Conradt, K. Daniilidis *et al.*, "Event-based vision: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 1, pp. 154–180, 2020.

[194] J. Jiao, H. Huang, L. Li, Z. He, Y. Zhu, and M. Liu, "Comparing representations in tracking for event camera-based slam," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1369–1376.

[195] L. Xu, C. Feng, V. R. Kamat, and C. C. Menassa, "An occupancy grid mapping enhanced visual slam for real-time locating applications in indoor gps-denied environments," *Automation in Construction*, vol. 104, pp. 230–245, 2019.

[196] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[197] H. Bay, T. Tuytelaars, and L. V. Gool, "Surf: Speeded up robust features," in *European conference on computer vision*. Springer, 2006, pp. 404–417.

[198] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," in *European conference on computer vision*. Springer, 2010, pp. 778–792.

[199] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International conference on computer vision*. Ieee, 2011, pp. 2564–2571.

[200] E. Karami, S. Prasad, and M. Shehata, "Image matching using sift, surf, brief and orb: performance comparison for distorted images," *arXiv preprint arXiv:1710.02726*, 2017.

[201] A. Bonarini, W. Burgard, G. Fontana, M. Matteucci, D. G. Sorrenti, and J. D. Tardos, "Rawseeds: Robotics advancement through web-publishing of sensorial and elaborated extensive data sets," in *In proceedings of IROS*, vol. 6, 2006, p. 93.

[202] J. McCormac, A. Handa, S. Leutenegger, and A. J. Davison, "Scenenet rgb-d: Can 5m synthetic images beat generic imagenet pre-training on indoor segmentation?" in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2678–2687.

[203] T.-M. Nguyen, S. Yuan, M. Cao, Y. Lyu, T. H. Nguyen, and L. Xie, "Ntu viral: A visual-inertial-ranging-lidar dataset, from an aerial vehicle viewpoint," *Internationl Journal of Robotics Research*, 2021.

[204] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The euroc micro aerial vehicle datasets," *The International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, 2016.

[205] X. Shi, D. Li, P. Zhao, Q. Tian, Y. Tian, Q. Long, C. Zhu, J. Song, F. Qiao, L. Song *et al.*, "Are we ready for service robots? the openloris-scene datasets for lifelong slam," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 3139–3145.

[206] W. Wang, D. Zhu, X. Wang, Y. Hu, Y. Qiu, C. Wang, Y. Hu, A. Kapoor, and S. Scherer, "Tartanair: A dataset to push the limits of visual slam," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 4909–4916.

[207] A. Handa, T. Whelan, J. McDonald, and A. J. Davison, "A benchmark for rgb-d visual odometry, 3d reconstruction and slam," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 1524–1531.

[208] E. Mueggler, H. Rebecq, G. Gallego, T. Delbruck, and D. Scaramuzza, "The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and slam," *The International Journal of Robotics Research*, vol. 36, no. 2, pp. 142–149, 2017.

[209] S. Wen, P. Li, Y. Zhao, H. Zhang, F. Sun, and Z. Wang, "Semantic visual slam in dynamic environment," *Autonomous Robots*, vol. 45, no. 4, pp. 493–504, 2021.

[210] Y. Yang, D. Tang, D. Wang, W. Song, J. Wang, and M. Fu, "Multi-camera visual slam for off-road navigation," *Robotics and Autonomous Systems*, vol. 128, p. 103505, 2020.

[211] H. Rebecq, T. Horstschaefer, and D. Scaramuzza, "Real-time visual-inertial odometry for event cameras using keyframe-based nonlinear optimization," in *British Machine Vision Conference*. University of Zurich, 2017, pp. 1–8.

[212] T. H. Nguyen, T.-M. Nguyen, and L. Xie, "Tightly-coupled ultra-wideband-aided monocular visual slam with degenerate anchor configurations," *Autonomous Robots*, vol. 44, no. 8, pp. 1519–1534, 2020.

[213] J. Sun, Z. Shen, Y. Wang, H. Bao, and X. Zhou, "Loftr: Detector-free local feature matching with transformers," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 8922–8931.

[214] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2961–2969.

[215] N. Wojke, A. Bewley, and D. Paulus, "Simple online and real-time tracking with a deep association metric," in *2017 IEEE international conference on image processing (ICIP)*. IEEE, 2017, pp. 3645–3649.

[216] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

[217] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[218] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[219] G. Jocher, A. Stoken, J. Borovec, L. Changyu, A. Hogan, L. Diaconu, F. Ingham, J. Poznanski, J. Fang, L. Yu *et al.*, "ultralytics/yolov5: v3. 1-bug fixes and performance improvements," *Version v3*, vol. 1, 2020.

[220] G. Liu, W. Zeng, B. Feng, and F. Xu, "Dms-slam: A general visual slam system for dynamic scenes with multiple sensors," *Sensors*, vol. 19, no. 17, p. 3714, 2019.

[221] J. Bian, W.-Y. Lin, Y. Matsushita, S.-K. Yeung, T.-D. Nguyen, and M.-M. Cheng, "Gms: Grid-based motion statistics for fast, ultra-robust feature correspondence," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4181–4190.

[222] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7263–7271.

[223] P.-Y. Tseng, J. J. Lin, Y.-C. Chan, and A. Y. Chen, "Real-time indoor localization with visual slam for in-building emergency response," *Automation in Construction*, vol. 140, p. 104319, 2022.

[224] E. Bonetto, P. Goldschmid, M. Pabst, M. J. Black, and A. Ahmad, "irotate: Active visual slam for omnidirectional robots," *Robotics and Autonomous Systems*, vol. 154, p. 104102, 2022.

[225] S. Wang, R. Clark, H. Wen, and N. Trigoni, "Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 2043–2050.

[226] E. Parisotto, D. Singh Chaplot, J. Zhang, and R. Salakhutdinov, "Global pose estimation with an attention-based recurrent network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 237–246.

[227] J. Czarnowski, T. Laidlow, R. Clark, and A. J. Davison, "Deepfactors: Real-time probabilistic dense monocular slam," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 721–728, 2020.

[228] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[229] X. Gu, Y. Wang, and T. Ma, "Dbld-slam: A deep-learning visual slam system based on deep binary local descriptor," in *2021 International Conference on Control, Automation and Information Sciences (ICCAIS)*. IEEE, 2021, pp. 325–330.

[230] V. Balntas, K. Lenc, A. Vedaldi, and K. Mikolajczyk, "Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors," in *CVPR*, 2017.

[231] C. Kerl, J. Sturm, and D. Cremers, "Dense visual slam for rgb-d cameras," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 2100–2106.

[232] ——, "Robust odometry estimation for rgb-d cameras," in *2013 IEEE International Conference on Robotics and Automation*, 2013, pp. 3748–3754.

[233] V. Ila, J. M. Porta, and J. Andrade-Cetto, "Information-based compact pose slam," *IEEE Transactions on Robotics*, vol. 26, no. 1, pp. 78–93, 2009.

[234] A. Li, J. Wang, M. Xu, and Z. Chen, "Dp-slam: A visual slam with moving probability towards dynamic environments," *Information Sciences*, vol. 556, pp. 128–142, 2021.

[235] E. Dong, J. Xu, C. Wu, Y. Liu, and Z. Yang, "Pair-navi: Peer-to-peer indoor navigation with mobile visual slam," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2019, pp. 1189–1197.

[236] L. Ma, C. Kerl, J. Stückler, and D. Cremers, "Cpa-slam: Consistent plane-model alignment for direct rgb-d slam," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 1285–1291.

[237] S. Han and Z. Xi, "Dynamic scene semantics slam based on semantic segmentation," *IEEE Access*, vol. 8, pp. 43 563–43 570, 2020.

[238] J. Zhao, Y. Huang, X. He, S. Zhang, C. Ye, T. Feng, and L. Xiong, "Visual semantic landmark-based robust mapping and localization for autonomous indoor parking," *Sensors*, vol. 19, no. 1, p. 161, 2019.

[239] M. Gonzalez, E. Marchand, A. Kacete, and J. Royan, "S3lam: Structured scene slam," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 6389–6395.

[240] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2," https://github.com/facebookresearch/detectron2, 2019.

[241] W. Wu, L. Guo, H. Gao, Z. You, Y. Liu, and Z. Chen, "Yolo-slam: A semantic slam system towards dynamic environment with geometric constraint," *Neural Computing and Applications*, vol. 34, no. 8, pp. 6011–6026, 2022.

[242] L. Nicholson, M. Milford, and N. Sünderhauf, "Quadricslam: Dual quadrics from object detections as landmarks in object-oriented slam," *IEEE Robotics and Automation Letters*, vol. 4, no. 1, pp. 1–8, 2018.

[243] M. Hosseinzadeh, K. Li, Y. Latif, and I. Reid, "Real-time monocular object-model aware sparse slam," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 7123–7129.

[244] Y. Fan, Q. Zhang, Y. Tang, S. Liu, and H. Han, "Blitz-slam: A semantic slam in dynamic environments," *Pattern Recognition*, vol. 121, p. 108225, 2022.

[245] L. Sun, J. Wei, S. Su, and P. Wu, "Solo-slam: A parallel semantic slam algorithm for dynamic scenes," *Sensors*, vol. 22, no. 18, p. 6977, 2022.

[246] Y. You, P. Wei, J. Cai, W. Huang, R. Kang, and H. Liu, "Misd-slam: multimodal semantic slam for dynamic environments," *Wireless Communications and Mobile Computing*, vol. 2022, 2022.

[247] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: an open-source library for real-time metric-semantic localization and mapping," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 1689–1696.

[248] D. Cai, R. Li, Z. Hu, J. Lu, S. Li, and Y. Zhao, "A comprehensive overview of core modules in visual slam framework," *Neurocomputing*, p. 127760, 2024.

[249] J. Yan, Y. Zheng, J. Yang, L. Mihaylova, W. Yuan, and F. Gu, "Plpf-vslam: An indoor visual slam with adaptive fusion of point-line-plane features," *Journal of Field Robotics*, vol. 41, no. 1, pp. 50–67, 2024.

[250] H. Pu, J. Luo, G. Wang, T. Huang, and H. Liu, "Visual slam integration with semantic segmentation and deep learning: A review," *IEEE Sensors Journal*, 2023.

[251] S. Koch, P. Hermosilla, N. Vaskevicius, M. Colosi, and T. Ropinski, "Sgrec3d: Self-supervised 3d scene graph learning via object-level scene reconstruction," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024, pp. 3404–3414.

[252] T. Schops, T. Sattler, and M. Pollefeys, "Bad slam: Bundle adjusted direct rgb-d slam," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 134–144.

[253] M. Grinvald, F. Furrer, T. Novkovic, J. J. Chung, C. Cadena, R. Siegwart, and J. Nieto, "Volumetric instance-aware semantic mapping and 3d object discovery," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 3037–3044, 2019.

[254] J. He, M. Li, Y. Wang, and H. Wang, "Ovd-slam: An online visual slam for dynamic environments," *IEEE Sensors Journal*, vol. 23, no. 12, pp. 13 210–13 219, 2023.

[255] P. Cong, J. Liu, J. Li, Y. Xiao, X. Chen, X. Feng, and X. Zhang, "Ydd-slam: Indoor dynamic visual slam fusing yolov5 with depth information," *Sensors*, vol. 23, no. 23, p. 9592, 2023.

[256] J. Han, R. Dong, and J. Kan, "Basl-ad slam: A robust deep-learning feature-based visual slam system with adaptive motion model," *IEEE Transactions on Intelligent Transportation Systems*, 2024.

[257] Y. Li, H. Zhao, X. Qi, L. Wang, Z. Li, J. Sun, and J. Jia, "Fully convolutional networks for panoptic segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 214–223.

[258] J. Hu, L. Huang, T. Ren, S. Zhang, R. Ji, and L. Cao, "You only segment once: Towards real-time panoptic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 17 819–17 829.

[259] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, "Voxblox: Incremental 3d euclidean signed distance fields for on-board mav planning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[260] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5828–5839.

[261] A. Tourani, S. Ejaz, M. Fernandez-Cortizas, J. L. Sanchez-Lopez, and H. Voos, "Visual s-graphs (vs-graphs) framework with ros wrapper," https://github.com/snt-arg/visual_sgraphs, 2023.

[262] A. Tourani, D. I. Avsar, J. L. Sanchez-Lopez, J. Lagerwall, and H. Voos, "imarker detector (ros version)," https://github.com/snt-arg/iMarker_detector_ros, 2022.

[263] A. Tourani, S. Ejaz, J. L. Sanchez-Lopez, and H. Voos, "Semantic scene segmentation framework with ros wrapper," https://github.com/snt-arg/scene_segment_ros, 2023.

[264] I. M. Chekam, I. Pastor-Martinez, A. Tourani, J. A. Millan-Romera, L. Ribeiro, P. M. Bastos Soares, H. Voos, and J. L. Sanchez-Lopez, "Robot suite," https://github.com/snt-arg/robot_suite, 2025.

[265] I. M. Chekam, I. Pastor-Martinez, A. Tourani, J. A. Millan-Romera, L. Ribeiro, P. M. B. Soares, H. Voos, and J. L. Sanchez-Lopez, "Interpretable robot control via structured behavior trees and large language models," *IEEE Access*, vol. 13, pp. 200 905–200 916, 2025.

[266] P. M. Bastos Soares, A. Tourani, M. Fernandez-Cortizas, A. Bikandi-Noya, H. Voos, and J. L. Sanchez-Lopez, "Smapper: A portable multimodal sensing platform," https://github.com/snt-arg/smapper, 2025.

[267] ——, "Smapper: A multi-modal data acquisition platform for slam benchmarking," *Journal of Intelligent & Robotic Systems*, vol. 112, no. 1, p. 20, 2026.